VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
ÚSTAV TELEKOMUNIKACÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
DEPARTMENT OF TELECOMMUNICATIONS

AUDIO/VIDEO STREAMING

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

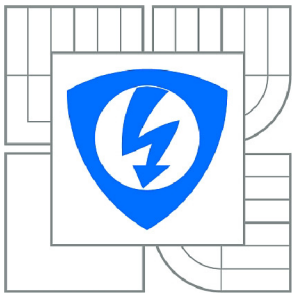
AUTOR PRÁCE
AUTHOR

Bc. RADEK ČENĚK

BRNO 2015

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

ÚSTAV TELEKOMUNIKACÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

DEPARTMENT OF TELECOMMUNICATIONS

AUDIO/VIDEO STREAMING

AUDIO/VIDEO STREAMING

DIPLOMOVÁ PRÁCE

MASTER'S THESIS

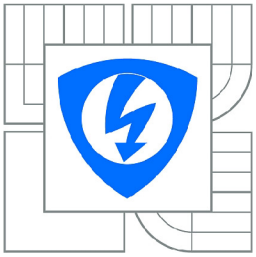
AUTOR PRÁCE — Bc. RADEK ČENĚK

AUTHOR

VEDOUCÍ PRÁCE — Ing. PETR ČÍKA, Ph.D.

SUPERVISOR

BRNO 2015



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky a komunikačních technologií

Ústav telekomunikací

Diplomová práce

magisterský navazující studijní obor
Telekomunikační a informační technika

Student: Bc. Radek Čeněk ***ID:*** 120769
Ročník: 2 ***Akademický rok:*** 2014/2015

NÁZEV TÉMATU:

Audio/Video streaming

POKYNY PRO VYPRACOVÁNÍ:

Prostudujte možnosti integrace audio/video streamovacího klienta do webové stránky pomocí WebRTC a HTML5. Vytvořte aplikaci (videotéku), díky které budete moci vybírat ze serveru jednotlivé pořady a streamovat je do Vámi vytvořeného klienta. Aplikace bude mít zabezpečenou sekci, ve které bude každý registrovaný uživatel moci spravovat nahrané videosekvence. Při nahrání videosekvence na server bude video automaticky překódováno do standardu WebM s předem definovaným rozlišením. Funkčnost aplikace ověřte běžnými způsoby. Zjistěte výpočetní náročnost serverové aplikace při kompresích různých typů vloženého videa.

DOPORUČENÁ LITERATURA:

[1] MANSON, Rob. Getting Started with WebRTC. London: PACKT PUBLISHING, 2013. ISBN 978-1782166306.

[2] GOOGLE. WebRTC [online]. 2011-2012 [cit. 2013-10-16]. Dostupné z: http://www.webrtc.org/

Termín zadání: 9.2.2015 ***Termín odevzdání:*** 26.5.2015

Vedoucí práce: Ing. Petr Číka, Ph.D.
Konzultanti diplomové práce:

doc. Ing. Jiří Mišurec, CSc.
Předseda oborové rady

UPOZORNĚNÍ:

ABSTRAKT

Tato práce čtenáře podrobně seznamuje s problémem audio/video streamingu skze síť Internet. Dále mu také přiblíží technologie MySQL, PHP, JavaScriptu a ffmpeg. Je zde uveden malý průzkum podobných aplikací. Při vytváření programu videotéky ukazuje složitost problému a jeho možné řešení. Součástí práce je i zjištění náročnosti komprese na server.

KLÍČOVÁ SLOVA

PHP, MySQL, JavaScript, audio, video, HTML5, protokol, HTTP, ffmpeg

ABSTRACT

This thesis introduces the reader in detail with the problem of audio / video streaming over the Internet. Introduces technologies MySQL, PHP, JavaScript and ffmpeg furthermore. There is little research which examines presented similar solutions. Creating video library program shows the complexity of the problem and its possible solution. The work also finding how is compression difficult for the server.

KEYWORDS

PHP, MySQL, JavaScript, audio, video, HTML5, protocol, HTTP, ffmpeg

RADEK, Čeněk *Audio/Video streaming*: diplomová práce. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací, 2015. 51 s. Vedoucí práce byl Ing. Petr Číka, Ph.D.

PROHLÁŠENÍ

Prohlašuji, že svou diplomovou práci na téma „Audio/Video streaming“ jsem vypracoval samostatně pod vedením vedoucího diplomové práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené diplomové práce dále prohlašuji, že v souvislosti s vytvořením této diplomové práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a/nebo majetkových a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

Brno ....

(podpis autora)

PODĚKOVÁNÍ

Rád bych poděkoval vedoucímu diplomové práce panu Ing. Petrovi Číkovi, Ph.D. za odborné vedení, konzultace, trpělivost a podnětné návrhy k práci.

Brno ....

(podpis autora)

Faculty of Electrical Engineering
and Communication
Brno University of Technology
Purkynova 118, CZ-61200 Brno
Czech Republic

http://www.six.feec.vutbr.cz

PODĚKOVÁNÍ

Výzkum popsaný v této diplomové práci byl realizován v laboratořích podpořených z projektu SIX; registrační číslo CZ.1.05/2.1.00/03.0072, operační program Výzkum a vývoj pro inovace.

Brno . . . . . .

. . . . . .

(podpis autora)

EVROPSKÁ UNIE
EVROPSKÝ FOND PRO REGIONÁLNÍ ROZVOJ
INVESTICE DO VAŠÍ BUDOUCNOSTI

OP Výzkum a vývoj
pro inovace

OBSAH

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

ÚVOD

Tato práce je zaměřená na audio/video streaming skrze síť Internet. Jsou zde popsány postupy a technologie, které byly využity během řešení práce.

První kapitola se věnuje průzkumu podobných projektů, přičemž jsou zde uvedeni čtyři zástupci, kteří přibližně splňují dané téma. U jednotlivých řešení je uvedený zkrácený popis, případně jejich výhody, či nevýhody.

Dále jsou popsány jednotlivé technologie, které jsou při řešení využity. Jedná se především o HTML5, PHP, databázový systém MySQL a program na konvertování videa ffmpeg. Jednotlivé části se zabývají daným okruhem.

Další kapitola se zabývá popisem a samotným řešením, jak byla aplikace tvořena. Jsou zde uvedeny části, ze kterých se řešení skládá a popsány jednotlivé funkce, které aplikace využívá.

Na závěr se práce zabývá měřením převodů videí různých formátů do mnou zvoleného formátu a závislostí způsobu zpracovávání na čase. Každé měření je zde popsáno spolu s grafy, které byly při měření zpracovány.

V přílohách jsou k dispozici nejdůležitější části kódu aplikace samotné a tabulka naměřených hodnot zjištěných při jednotlivých měření.

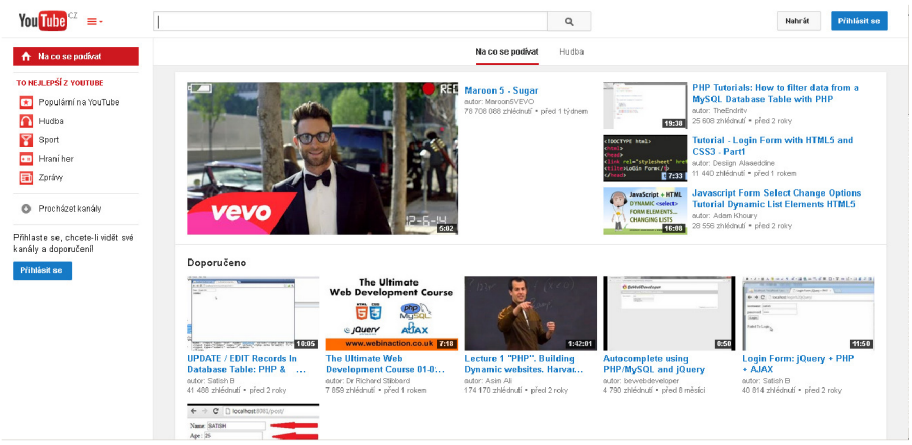
1 PRŮZKUM PODOBNÝCH PROJEKTŮ

Na Internetu je možné najít mnoho podobných projektů, jako je řešení uvedené v této práci. Každý z uvedených projektů je specifický, ale v některých aspektech lze naleznout jistou výraznou podobnost. Největším rozdílem bývá obsah, jenž je na dané stránce přehráván nebo archivován. Nejvíce je takovýto systém využíván u stránek s omezeným přístupem tzv. stránek pro „dospělé“. Díky databázím si udržují informace o uživatelích a jejich potenciálních platbách. Zásluhou toho zjišťují správci kolik obsahu si může daný zákazník ještě přehrát.

Stránky jsou často provázány se sociálními sítěmi, převáženě proto, aby se zvýšila jejich návštěvnost a došlo tak i k nenásilné reklamě. Obecně platí, že s návštěvností roste i server i jeho požadavky, proto se dá sehnat i méně známý projekt s menším obsahem. O některých zástupcích si povíme dále.

1.1 Youtube.com

Jedná se největší a nejznámější webové stránky, které jsou zaměřeny na online streaming. Jednotlivá videa jsou nahrávána uživateli, kteří se musí přihlásit pomocí google účtu. Po nahrání na server se můžeme na dané video podívat pomocí odkazu či využít vyhledávání. To funguje tak, že se vyhledává nejvhodnější vyhledávaný výraz. Výsledek s posterem a popisem jsou vypsány pod sebe jako dvacet nejvhodnějších výrazů. Server žije díky reklamě, která se interaktivně vkládá do videí. Zásluhou toho mohou i autoři videí vydělávat přehráváním jejich videí.



Obr. 1.1: Hlavní stránka www.youbube.com

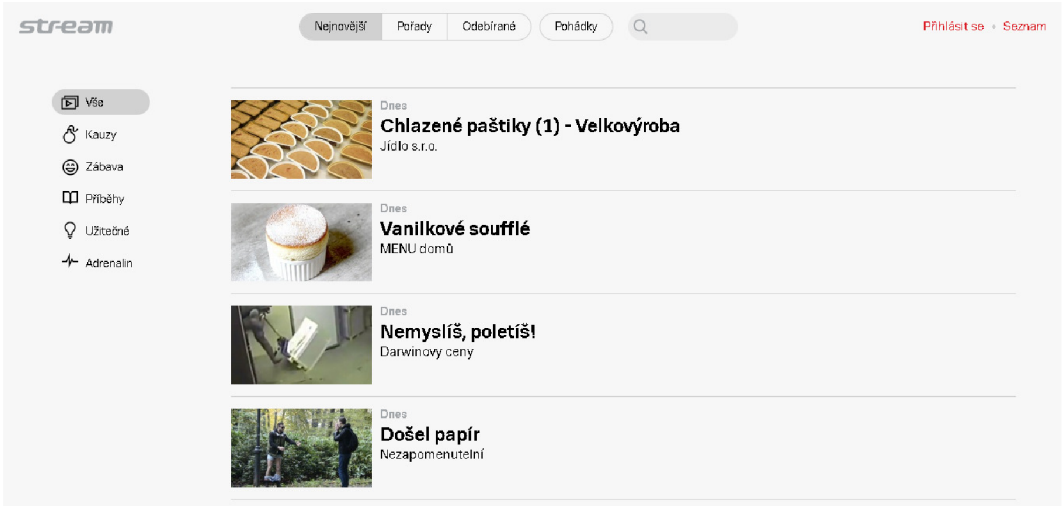
1.2 Zvraceny.cz

Jedná se o jeden z větších projektů v České republice. Po registraci je možné vkládat videa. Jednotlivé klipy jsou uloženy na server a přidávány do databáze s určitým zpožděním. Je zde šest hlavních kategorií a téměř každý si zde najde svůj šálek kávy.

Obr. 1.2: Přijmutí podmínek www.zvraceny.cz

1.3 Stream.cz

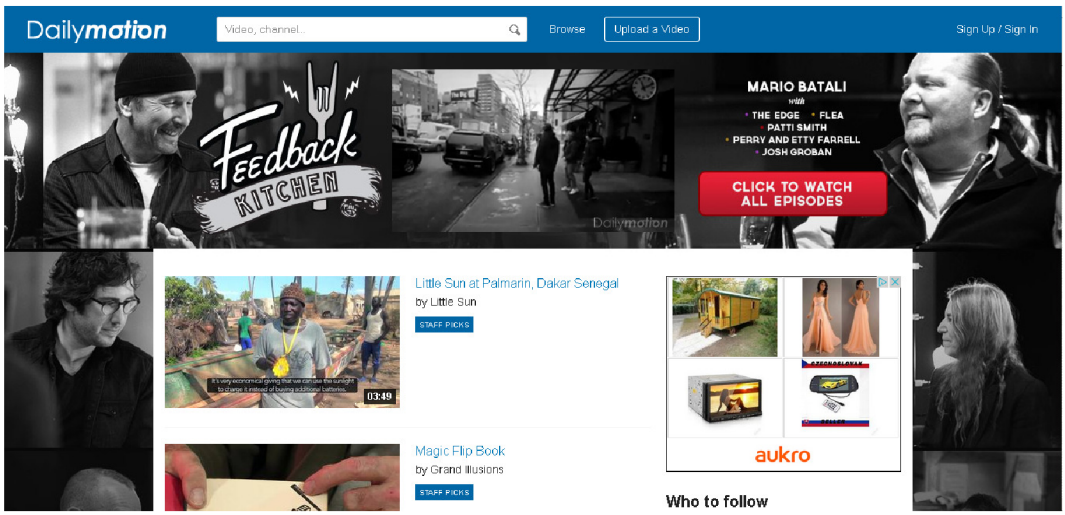
Další český zástupce se nazývá Stream.cz. Nyní jde o online televizi. Skládá se z jednotlivých programů, které je možné si přehrát. Uživatelé se mohou přihlásit k odběru novinek v pořadech, které uživatele zajímají. Bohužel zde není možnost volnosti uživatelů a tak je tento projekt držen ve svých kolejích.



Obr. 1.3: Internetová televize Stream.cz

1.4 Dailymotion.com

Další z řady video serverů pro přehrávání a sdílení videí je tento francouzský projekt. V zahraničí je velmi populární a v posledních letech získává renomé i v České republice. Obsahuje velké množství videí a pořadů, které je možné sledovat. Některé české stránky zde mají úložiště pro svoje videa. Nevýhodou je velmi vlezlá reklama, která se opakuje po každých pěti minutách. Vkládat filmy však může po registraci každý, a tak je k dispozici opravdu velké množství videí.

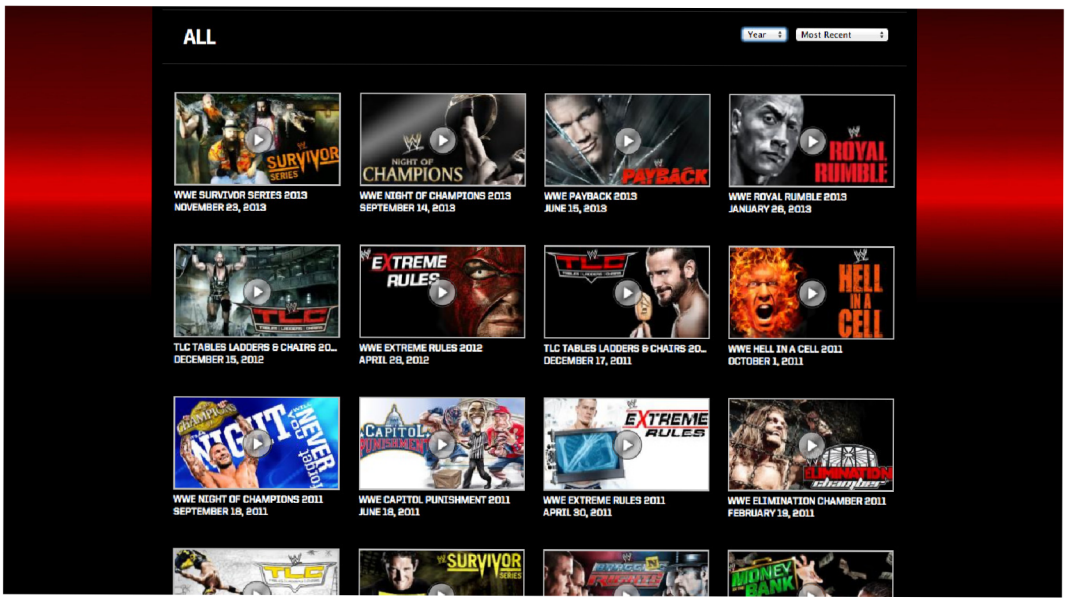


Obr. 1.4: Video úložiště Dailymotion.com

1.5 Ostatní

Jak již bylo řečeno, podobných projektů je opravdu velmi mnoho. Jeden z největších provozovatelů je odvětví porno průmyslu. Zde je účelové šíření videí za účelem výdělku, a protože je toto odvětví v době Internetu ještě více žádoucí, roste i počet takovýchto provozovatelů.

V roce 2014 rozjela jedna z největších zábavných společností ve spojených státech WWE svojí vlastní síť s názvem WWE Network, kde jsou uloženy všechny zápasy a živě přenášeny všechny show, které se dějí v reálném čase. Jde opět o výhradně předplatitelský program.



Obr. 1.5: WWE network

2 POUŽITÉ TECHNOLOGIE

Aby práce mohla vzniknout, musel jsem využít mnoho novinek ve světě programování webových stránek a skloubit je spolu s novými postupy či řešeními, abych mohl splnit dané zadání. Jednotlivé technologie jsou uvedeny právě v této kapitole.

2.1 HTML5

HTML5 je nový značkovací jazyk pro programování webových stránek nové generace. Jeho konečná podoba specifikace byla vydána až 28. října 2014. Od předchozích verzí se rozšířila základna nových prvků, které je možné využít pro samotnou tvorbu. Největším rozdílem od předchozích verzí je podpora multimediálního přehrávání již v základní verzi jazyku. V kapitole je zmíněn pouze zlomek informací, které je možné naléznout v literatuře, například HTML5: programujeme moderní webové aplikace [1] a Tvorba internetových stránek pomocí HTML, CSS a JavaScriptu [2].

2.1.1 Změny v HTML5

Jako největší novinka, která bude také nejvíce využita v této práci, je zjednodušená práce s multimediálním obsahem. Novinky se týkají jak videa a audia, tak i zobrazování obrázků pomocí prvku `<canvas>`.

Většina dalších novinek je zaměřena na zlepšení organizace stránek, přidává nové funkce a zlepšuje práci s formulářem. Vybrané nové tagy jsou uvedeny v tabulce 2.1.

Obr. 2.1: HTML5 logo

2.1.2 Video

Tento prvek je pro tuto práci stěžejní. Díky němu lze totiž realizovat přehrávání filmů, spotů nebo dokonce vytvořit komunikační rozhraní. Možnosti tohoto prvku jsou veliké. Nejdůležitější označení prvku je `ID`, díky tomu je možné jednotlivé prvky mezi sebou rozeznat. Mnoho z nastavení je společné i pro prvek `<audio>`, jako je například autoplay, src atd. Důležitým nastavením je šířka a výška, která určuje

Tag	Použití
<article>	Jedná se o příkaz pro rozdělení stránky. Vytvoří samostatnou a nezávislou část, jako je například článek nebo komentář.
<aside>	Definuje libovolný obsah, který by měl být spojen s hlavní stránkou. Lze tedy využít například pro boční panel na stránce.
<canvas>	Jde pouze o plátno pro dynamické vykreslování obrázků, grafiky, her, grafů a podobně. Element je tedy jen kontejner pro grafiku.
<details>	Příkaz je určený k popisu podrobností, které se týkají dokumentu či jeho částí.
<footer>	Jedná se o patičku, která lze využít buď u článků k popisu autora, nebo uvedení odkazů. Je možné použít i pro vytvoření patičky celé stránky. Lze zde uvést údaje, jako jsou informace o stránce, či autorských právech.
<header>	Obdobně jako patička i hlavička může být použita pro označení článku či označení celé stránky.
<mark>	Představuje zvýraznění nebo označení určitého textu.
<nav>	Zde je prostor pro vytvoření navigačního rozhraní pro stránku neboli menu.

Tab. 2.1: Nové tagy zahrhuté v HTML5

velikost videa. Ta by se měla nastavit primárně, jinak se video samo nastaví na velikost, ve kterém je originálně nahráno. Možnosti nastavení jsou uvedeny níže.

```
<video autoplay id="zvuk" src="zdroj" controls></video>
```

Obdobně jako u prvku `<audio>` je nutné i zde zajistit kompatibilitu videa s internetovým prohlížečem. Možnost mnohonásobného zdroje řeší tento problém. Následující tabulka 2.2 zobrazuje, které prohlížeče můžeme použít k vybraným formátům videa.

Prohlížeč	WebM	MP4	OGG
Opera	ano	(od Opera 25)	ano
Firefox	ano	ano	ano
Safari		ano	
Chrome	ano	ano	ano
Internet explorer		ano	

Tab. 2.2: Podporované formáty prvku <video>

Příklad mnohonásobného zdroje, který je uveden níže, ukazuje, jak vyřešit kompatibilitu videa. Pokud prohlížeč nepodporuje ani jeden ze zdrojů, ukáže se místo videa zobrazí chybová hláška.

```
<video width="320" height="240" autoplay>
    <source src="video.mp4" type="video/mp4">
    <source src="video.ogg" type="video/ogg">
    <source src="video.webm" type="video/webm">
    Váš prohlížeč nepodporuje dané video.
</video>
```

2.2 MySQL

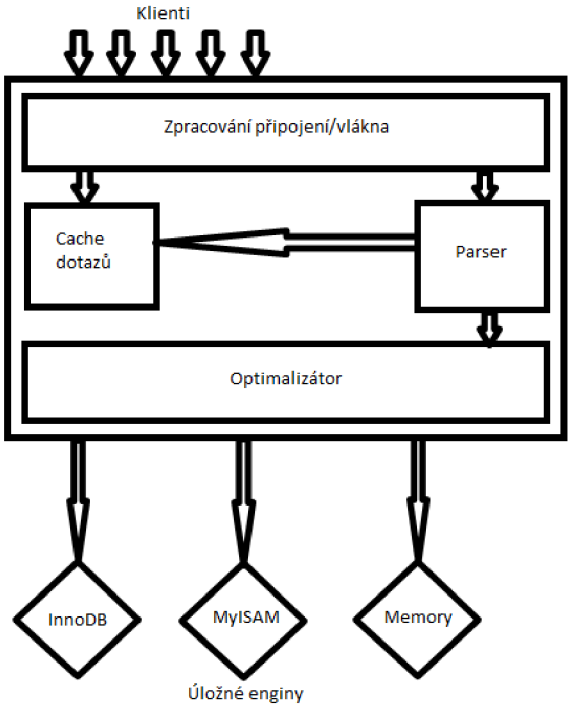
MySql je databázový systém, který byl vytvořen firmou MySQL AB. Systém je možný provozovat pod bezplatnou licencí GPL nebo pod placenou variantou. Databáze jako taková má základ v jazyce SQL, který poskytuje spojení s databází. Praktické znalosti tohoto systému předkládá například Dubois [5]. MySQL lze provozovat na jakémkoliv operačním systému, ať se jedná o Linux, Microsoft či jiný systém. Hlavní předností MySQL oproti jiným databázovým systémům je její rychlost. Té bylo podřízeno vše, proto ještě nedávno nepodporovala pohledy, triggery a uložené procedury. Uživatelům však tyto funkce chyběly, proto byly v posledních letech doplněny. Nejčastěji volenou kombinací pro základní webový server se tak stalo spojení Linux, MySQL, PHP a Apache.

Obr. 2.2: MySQL logo

2.2.1 Architektura MySQL serveru

Server MySQL se skládá ze tříúrovňového modelu, čímž se odlišuje od jiných databázových systémů. Vrchní vrstva obsahuje procedury, které se starají o připojení uživatelů k serveru a řízení provozu. Prostřední vrstva je zaměřená na samotné řízení serveru. Jsou zde například funkce pro rozbor, analýzu či optimalizování. Poslední vrstva obsahuje úložné enginy. Server komunikuje s těmito enginy pomocí API za účelem vytvoření dat, mazání nebo vyhledávání. Díky tomu vytvářejí velmi transparentní ovládání databáze. Zmiňovaná API v sobě obsahuje spoustu funkcí,

na které volané enginy odpovídají. Zásluhou toho enginy nevytvářejí žádný provoz, pouze odesílají odpovědi. Celá architektura je pak vidět na obr. 2.3.



Obr. 2.3: Architektura MySQL serveru

2.2.2 Správa připojení a bezpečnost

Každý klient, který se připojí, dostane uvnitř serverového procesu vlastní vlákno. Následné dotazy jsou řešeny pouze v rámci tohoto vlákna, které běží na daném jádru. Vlákna jsou serverem udržována v krátkodobé paměti (cache), proto nemusí být ukončována pro každé spojení. Autentizace je založena na kontrole hostitele, uživatelského jména a hesla. Při připojení se tak musí zadávat jméno, heslo, databáze a tabulka, ke které se chceme připojit, jinak je spojení přerušeno serverem. Server jako takový nejen ověřuje pravost uživatele, ale kontroluje také jeho práva pro připojení.

2.2.3 Optimalizace a vykonávání

Systém se zabývá nejen zpracováním dat a příkazů, ale rovněž optimalizací. Dělá to hlavně proto, aby ušetřil strojový čas při prohledávání velkého množství dat. Provádí proto rozbor dotazů a díky tomu, pak může vytvořit interní stromovou strukturu. S její pomocí může provádět libovolné procesy pro optimalizaci. Pomocí

klíčových slov (hints) může programátor optimalizátoru ovlivnit rozhodovací proces. Konečné slovo má přesto úložný engine. Ten totiž určuje, jak server optimalizuje dotaz. Optimalizátor je enginem upozorněn, jakou výbavu funkcí má a informuje ho o nákladech na jednotlivé operace, statistikách a datumech tabulek. Poté se podívá do query cache, kde může mít uložené příklady `SELECT` se sadami, které byly při příkazu užity. Pokud tam podobný záznam není uložen, tak server provede rozbor. Jestliže však záznam existuje, tak optimalizátor nic nevykoná, pouze předá již uloženou výslednou sadu.

2.2.4 Úložné enginy (úložiště dat)

Každá databáze je na MySQL server uložena do podadresáře na odkladovém souborovém systému. Jednotlivé tabulky jsou uloženy jako definice v souborech typu `.frm`, jež má stejný název jako tabulka samotná. Protože je název tabulky provázán se souborem, je vhodné dodržovat velikost písmen, jelikož unixové systémy dodržují velikosti písma, což by mohlo znamenat problém při přechodu z windowsové platformy právě na unixovou platformu.

Úložné enginy jsou velmi podobné modulům. K distribuci daného serveru je možná jejich instalace a tím i možnost dotvoření serveru podle představ programátora. Jejich aktuální seznam je možné zjistit pomocí příkazu `SHOW ENGINES`. Jednotlivé enginy se liší svými možnostmi a použitím, jak nám ukazuje tab. 2.3.

Úložný engine	Popis
ARCHIVE	Engine uzpůsobený pro ukládání velkého množství neindexovaných dat.
BLACKHOLE	Engine, který data přijímá, ale neukládá je (zahazuje je).
CSV	Ukládá data v textovém formátu CSV.
EXAMPLE	Nefunkční engine, který slouží jako ilustrační pro potřeby zdrojových kódů databáze MySQL a využijí ho tedy jen její vývojáři.
FEDERATED	Engine vytváří místní tabulku, která zastupuje tabulku odjinud.
InnoDB	Engine zpracováná mnoho krátkodobých transakcí.
MEMORY	Vysoce výkonný engine, který data uchovává pouze v operační paměti, při restartu serveru jsou data ztracena.
MERGE	Sloučení dat z několika MyISAM tabulek o stejné struktuře, starší alternativa k partition.
MyISAM	Engine ukládá tabulky ve dvou souborech (data a indexy). Jako nejstarší engine podporuje mnoho funkcí.

Tab. 2.3: Jednotlivé úložné enginy

2.3 PHP

PHP je skriptovací programovací jazyk, který pracuje na straně serveru, tudíž všechny jeho příkazy vykonává server a k uživateli je přenesen pouze výsledek. Vznikl v roce 1996 a nyní se jedná o nejrozšířenější programový jazyk v rámci vytváření webových stránek a aplikací.

Je využívám především k programování dynamických internetových stránek a webových aplikací. Jazyk podporuje všechny platformy a programátor se tak nemusí starat o zvláštnosti jednotlivých platforem. PHP podporuje, podobně jako JavaScript, mnoho účelových knihoven pro zpracování textu, obrázků, grafiku, práci se soubory, přístup k databázím či pro podporu různých internetových protokolů.

2.3.1 Možnosti PHP

PHP je velmi důležitý pomocník pro vývoj webových stránek. S jeho pomocí lze vytvořit téměř všechen dynamický obsah stránky, jako je diskusní fórum, počítadlo, kniha návštěv, anketa, graf nebo jiné. Nejlépe nás se základy seznámí literatura PHP nejen pro začátečníky [3] nebo rozšířenější PHP a MySQL: bez předchozích znalostí [4]. Největší jeho předností je možnost propojit své stránky s databázemi, jako je například MySQL.

Nejčastější užití PHP kódu je však při používání tzv. šablon. Ta vypadá tak, že se vytvoří kostra stránky, do které se vkládají opakující se části, jako je například patička, menu nebo hlavička. Jednotlivé části jsou umístěny ve zvláštních souborech a ty se pak vkládají do šablony podle toho, které chceme využít. Změnit po nějaké době menu, tak nebude problém. Stačí změnit pouze jeden soubor a ne každý zvlášť.

Obr. 2.4: PHP logo

2.3.2 Instalace

Protože PHP je jazyk, kterému nestačí pouze webový prohlížeč (akce provádí server), je nutné jej nainstalovat do svého počítače. Nejčastěji je využíván webový server Apache, který je doplněn o instalaci PHP, která zajistí podporu tohoto jazyka

na serveru.

Webová stránka s prvky PHP má nejčastěji koncovku `.php`. Avšak je možné použít i koncovky `.php3`, `php4`, `php5` a `phtml`. Pokud však použijeme koncovku s číslovkou, tak bychom měli pamatovat na to, že při vydání nové verze může námi vytvořená stránka vypadat zastarale.

2.3.3 Webhosting

Ne každý webhosting zahrnuje podporu PHP. Potřebná podpora je u webhostingu nadstandardní službou za příplatek. Nicméně lze i free webhosting sehnat. Při výběru webhostingu pro PHP stránky si pečlivě přečtěte, co nabídka zahrnuje.

2.4 JavaScript

Jedná se o objektově orientovaný skriptovací jazyk, který byl standardizován již v roce 1997 asociací ECMA a o rok později i asociací ISO. JavaScript jako takový nemá nic společného s jazykem Java, kromě podobnosti jeho syntaxí. Jedná se o multiplatformní programovací jazyk, který je nejčastěji využíván pro dynamizaci WWW stránek a ovládání aktivních prvků. Z velké části se vkládá jeho kód přímo do HTML kódu, ale může být vytvořen i externí skriptovací soubor. Daná problematika je prakticky popsána v [2].

Program napsaný v JavaScriptu se spouští výhradně až po načtení stránek ze serverů, kde jsou uloženy. Teprve po načtení může pracovat s funkcemi, které jsou vykonávány výhradně na straně klienta. Jednotlivé funkce mohou využívat i jiné programovací jazyky, které jsou k dispozici na straně serveru, jakou je například PHP.

Kromě využití na webu je JavaScript využíván i pro rozšíření některých aplikací jako Adobe Acrobat nebo je možné využít soubory psané v tomto jazyce jako náhradu za stávající dávkové soubory MS-DOS. V dnešní době jsou vytvořeny i aplikace, které umožňují práci JavaScriptu na straně serveru, je v něm napsán server samotný. Jedná se tak například o aplikace Node.js či Apache.

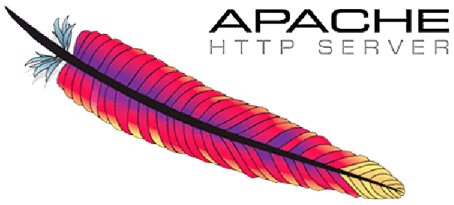
3 POUŽITÉ SOFTWAROVÉ VYBAVENÍ

Jako použitý software jsou uvedeny tři hlavní aplikace, které jsou potřeba pro vytvoření multimediálního webového serveru. Hlavní aplikace se jmenuje Apache. Jedná se o webový server, na kterém jsou uložena všechna data, která jsou využívána pro potřeby videotéky. Dále je zde dvojice programů, které zajišťují běh databáze. Jedná se o MySQL workbench, díky kterému je možné vytvořit databázi a MySQL community server, který zajišťuje běh databázového serveru.

3.1 Apache

Jedná se o softwarový webový server s otevřeným kódem, který je možný instalovat na většinu aktuálních operačních systémů.

Hlavní vývoj programu začal v roce 1993, kdy se na svět dostala první verze programu na Illinoiské univerzitě. Bohužel se na projektu přestalo pracovat, i když se stal mezi správci webových serverů populární. Ti se pak snažili o zkvalitnění služeb, a tak většina z nich vydávala své patche. V roce 1995 byla založena e-mailová konference za účelem sjednocení úprav, aby se dále mohl distribuovat celý balíček těchto posbíraných úprav. Ve stejném roce byla vydána první veřejná a upravená verze. Dnešní verze nemá již nic společného s prvotinou z roku 1993 a o její inovace se stará Apache group.



Obr. 3.1: Apache logo

Od roku 1996 se stal Apache jedním z nejpopulárnějších serverů na Internetu. V dnešní době má více než 50% zastoupení, jak ukazuje výzkum April 2014 Web Server Survey[1]. Server podporuje mnoho funkcí, jež jsou implementovány jako kompilované moduly, které rozšiřují jádro. Jedná se o funkce například pro podporu programovacích jazyků na straně serveru, funkce pro bezpečnost či kompresní funkce pro snížení objemu dat. Díky virtuálnímu hostingu je možné uskutečnit obsluhu více stránek na jednom fyzickém počítači.

[1]Průzkum řešení webových serverů z dubna 2014.
Dostupné z: http://news.netcraft.com/archives/2014/04/02/april-2014-web-server-survey.html

Hlavní předností Apache je poskytování mnoha MultiProcessing modulů. Zásluhou toho může správce optimalizovat server přímo pro své požadavky, aniž by bylo nutné spuštění částí, které nevyužije. Díky správnému nastavení může správce dosáhnout velmi nízké latence a co nejvyšší propustnosti. Nejvýkonnější verze z toho pohledu je více vláknová verze, která paralelně řeší větší počet operací ve stejném čase.

3.2 MySQL Workbench

MySQL Workbench je program pro vizuální návrh struktury databáze MySQL. Spolu s MySQL community serverem vytváří vhodnou instalaci pro správu a chod databázového serveru. Mezi hlavní úkony pro workbench patří:

- návrh a modelování databáze,
- administrace databáze,
- migrace databáze.

Obr. 3.2: MySQL workbench okno při načítání

3.3 MySQL community server

Jedná se o celosvětově nejoblíbenější open source projekt pro databázové systémy. Systém jako takový, je podpořen aktivním zapojením vývojářů a nadšenců. Aplikace nám umožní zprovoznění databázového serveru, který však potřebuje naprogramování. K tomuto účelu může programátor využít buď dceřiného programu MySQL Workbench nebo si databázi vytvořit i pomocí jiných dostupných programů.

3.4 FFMPEG

3.4.1 Historie

Zakladatelem projektu byl Fabrice Bellard v roce 2000. Mnoho nynějších vývojářů ffmpeg je také součástí projektu MPlayer. Jak již název naznačuje, projekt vychází ze skupiny kodeků MPEG. Následné přízvisko „FF“ znamená fast forvard. Hlavní operační systém, ve kterém byl vyvíjen, je Linux, ale nyní může být provozován pod kterýmkoliv operačním systémem, jak mac OS, AmigaOS, Windows aj. Kromě mnoha operačních systémů podporuje ffmpeg i většinu známých architektur.

Verze 0.5 se objevila po dlouhé době bez formálního vydání. Po tomto představení se vydávaly průběžně nové verze každé tři měsíce. Během vývoje byly využívány dva hlavní kodeky a jeden video kontejner. V létě 2010 oznámil vývojový tým sestavení ffvp8 dekodéru, který byl podle následných testů rychlejší než již používaný libvpx Google. I díky tomu podporuje ffmpeg od verze 0.6 i WebM a VP8.

V říjnu 2010 je ffmpeg umístěn přímo do operačních systémů Ubuntu a Debian. O rok později nastupuje podpora platformy Windows. Následují roky, kdy byla řešena podpora dalších kodérů a dekodérů. Jedněmi z posledních byly OpenHEVC a High Efficienty Video Coding. Na začátku roku 2014 oznámili zaměstnanci Google že bylo opraveno více než tisíc chyb, které ffmpeg obsahoval. V dnešní době obsahuje balík ffmpeg mnoho součástí, které jsou uvedeny níže.

- Nástroje
  - Ffmpeg je utilita pro příkazový řádek pro konverzi video formátů. Podporuje také grabování a kódování v reálném čase z TV karty.
  - Ffserver je HTTP a RTSP multimediální streamovací server pro živá broadcastová vysílání. Podporuje rovněž posun času.
  - Ffplay je jednoduchý multimediální přehrávač založený na SDL a knihovnách ffmpeg.
  - Ffprobe je jednoduchý analyzátor multimediálních streamů.
- Knihovny
  - Libswresample je knihovna, která umožňuje převzorkování audio stop.
  - Libavcodec je nejdůležitější knihovna, protože obsahuje všechny audio/video dekodéry.
  - Libavformat obsahuje muxery a demuxery.
  - Libavutil je využívána programem ffmpeg, aby volala jednotlivé nástroje.
  - Libpostproc obsahuje rutiny pro postprocesing videa.
  - Libswscale umožňuje změnu rozlišení videa.

3.4.2 ffmpeg

Ffmpeg je velmi rychlý audio/video konvertor a může také zachytávat živé zdroje audia/videa. Pracuje v příkazovém řádku a je možné jej využívat na všech platformách operačních systémů. Jednotlivá videa může konvertovat mezi libovolnými typy s libovolnými vzorkovacími frekvencemi, měnit velikost videa nebo jejich rozlišení. Jako zdroj pro ffmpeg může posloužit téměř cokoliv od jednoduchého souboru až po síťové vysílání. Jednotlivé soubory mohou být zpracovány do jednoho či několika výstupních entit.

Každý vstupní nebo výstupní soubor může v zásadě obsahovat libovolný počet proudů různých typů (video, audio, titulky, přílohy, data). Povolený počet toků nebo jejich typy mohou být omezeny formátem kontejnerů. Volba proudů, které budou zpracovány, se děje buď automaticky, nebo nastavením uživatele pomocí indexů.

Můžeme využít i vložení několika vstupních souborů, ale musí se správně definovat postup převodu videa, jinak program nemusí pracovat korektně.

3.4.3 ffprobe

Utilita ffprobe shromažďuje informace z multimediálních souborů a zobrazuje je v čitelné formě. Obdobně jako u ffmpeg pracuje v příkazové řádce a jednotlivé výpisy jsou zde rovněž zobrazeny. Může být například použit pro kontrolu datových toků multimédií nebo zjištění, o jaký typ se právě jedná.

Pokud je na vstupu zadán název souboru, tak se jej ffmprobe pokusí otevřít a prozkoumat jeho obsah. Pokud zjistí, že se jedná o multimediální soubor, tak jej zpracuje a jako výsledek vypíše obecné informace. Ffprobe může pracovat jako jednoduchý příkaz, ale dá se definovat, co má ve zkoumaném souboru hledat.

Video soubory často obsahují více stop, jako je zvuková stopa a video stopa. Ty jsou od sebe odděleny, a pokud chce uživatel o jedné z těchto stop vědět více informací, tak musí svůj požadavek konkretizovat pouze na danou stopu.

4 VIDEOTÉKA

V rámci řešení diplomové práce se měla vytvořit jednoduchá aplikace, kde by bylo možné přehrávat videa a nahrávat je na server. Jako vzorový příklad bylo zvoleno téma videoklipů. Hlavním důvodem volby byla relativně krátká doba konverzí jednotlivých klipů na požadovaný formát `.webm`. Samozřejmě je možné použít i jiná videa, avšak jejich příprava pro vložení na webový server je časově delší. Následující stránky nastiňují možné řešení, které je dostupné na přídavném médiu, kde jsou k dispozici také programy nutné pro korektní fungování aplikace samotné.

Řešení se dá rozdělit na tři hlavní části, o kterých je napsáno níže. Jedná se především o práci s videem samotným, možnosti jeho uskladnění, správu nebo úpravu. Hlavní část aplikace je uložena na serveru Apache s podporou PHP. Další a nedílnou součástí je server MySQL, který se stará o databázi záznamů a uživatelů.

Pro inspiraci a nápad, jak realizovat aplikaci, jsem využil PHP 5 a MySQL 5: hotová řešení [6].

4.1 Klient

Klientská část je tvořena PHP aplikací. Výhodou toho řešení je nejen jeho rychlost a účinnost, ale hlavně to, že nikdo nezjistí, co naše soubory PHP přesně obsahují, protože uživateli se zobrazí zdrojový kód jako klasická HTML stránka. Aplikace samotná se skládá z hlavní stránky, jinak také šablony, a do té jsou vloženy další části. Při pohledu na screenshot aplikace 4.1 si můžeme všimnout pěti oddílů.

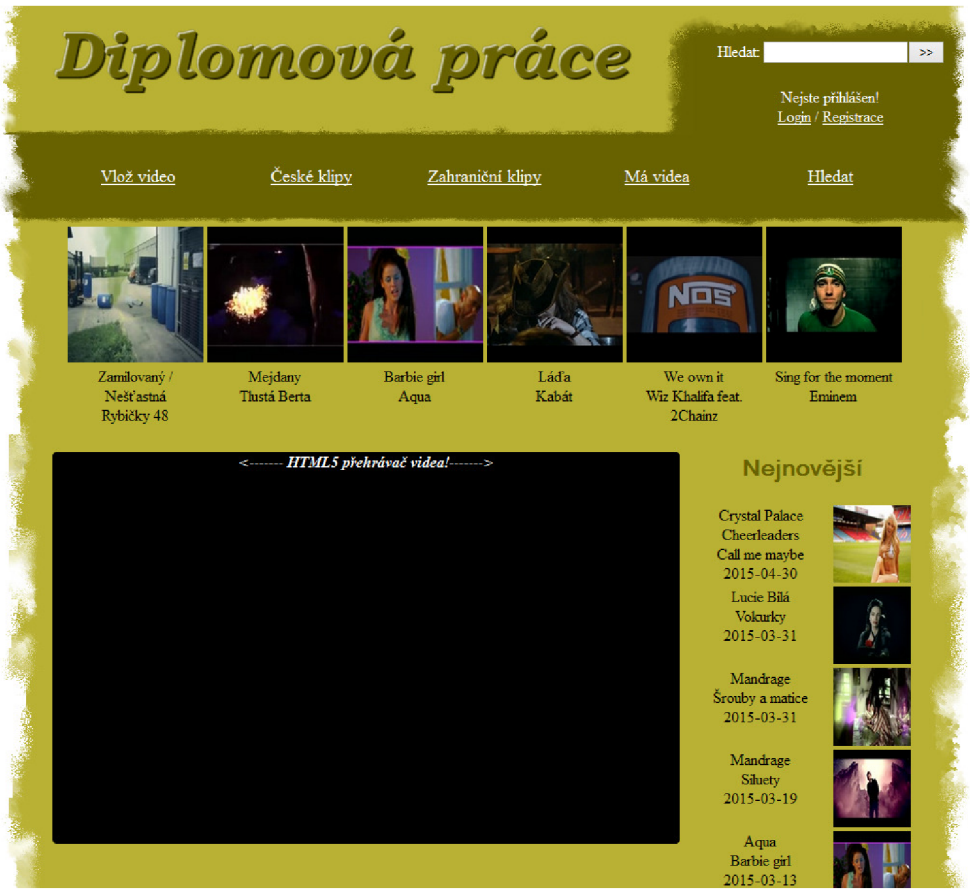
Většina těchto částí je dynamická a podle uživatele se mění kromě části s oddílem pro menu, kde se pouze kontroluje na určitých odkazech, zda je uživatel zalogovaný. O logování se stará část v pravém horním rohu stránky. Pod ní se nachází prostor, kde je umístěno horizontální menu. Níže je umístěn prostor pro náhodně generovaná videa.

Zbylé dvě části se mění nejvíce podle uživatele. Vlevo je na hlavní stránce umístěn přehrávač, avšak během práce se stránkou se zde objevují také tabulky pro registrování uživatele, vkládání videí aj. Napravo je sloupec, kde se zobrazují nejnovější videa, ale je zde možné vidět i zahraniční nebo české klipy.

4.2 Video

4.2.1 Přehrávání videa

Při vytváření jednoduchého přehrávače jsem využil rozšíření HTML5, které umožňuje přehrávat videa pomocí tagu `<video>`. Aplikace je zaměřená na největší část



Obr. 4.1: Aplikace na straně klienta

uživatelů, kteří využívají prohlížečů Firefox, Chrome či Opera. Tento výběr jsem použil proto, že již neznám nikoho, kdo by využíval služeb Internet Exploreru nebo Safari. I zde by práce mohla být funkční, pokud by si uživatel doinstaloval jednoduchou utilitku, která umožňuje přehrávání mnou zvoleného formátu videa.

Přehrávač samotný jsem obalil do divu přehrávač proto, abych mohl vytvořit stavový řádek, který uživatele informuje o přehrávaném videu (skladbě). Jako zdroj videa jsou klipy ve formátu `.webm`. Mohli bychom nabídku rozšířit i o formát `.mp4` a aplikace by fungovala na všech prohlížečích. Z důvodu nutnosti dvojité konverze jsem se pro tuto variantu nerozhodl.

4.2.2 Tvorba a konverze videa

Tvorba videa začíná u uživatele samotného. Ten je považován za zdroj informací. Vybere si klip, který chce nahrát na stránku a tím iniciuje program pro konverzi. Pokud si sám nevybere úvodní obrázek pro video, tak se vytvoří náhled sám z videa na dvacáté sekundě klipu. Uživatel dále nastaví název, interpreta a viditelnost videa.

Po zadání vstupních informacích program zjistí, jaké má video rozlišení a podle koeficientů 4:3 nebo 16:9 nastaví nejbližší možné rozlišení podobné originálnímu klipu. Při konverzi se vezme nejbližší hodnota šířky nebo výšky a podle té se pak pokračuje dále. Druhá hodnota velikosti se poté nemění a program videu přidá černé pruhy tak, aby nedošlo ke změně videa. Informace z videa se dají zjistit pomocí ffprobe, přičemž musí být známo umístění této utilitky, kterou obsahuje ffmpeg samotný. Celé nastavení, ve kterém je kromě rozlišení i cílová složka, se odešle na zpracování právě programu ffmpeg.

Variant, jak správně nastavit tento program, je mnoho. Pro můj případ mi připadala nejvhodnější varianta, která je uvedena níže.

```
$ffmpeg -i $videoFile -codec:v libvpx -quality good -cpu-used 5
-b:v 500k -qmin 10 -qmax 42 -maxrate 500k -bufsize 1000k -threads 4
-vf scale=$scale -codec:a libvorbis -b:a 128k C:\\server\\www\\video\
\$typ\\$uloz.webm";
```

V příkazu se vyskytuje několik proměnných jedná se především o `$ffmpeg`, ve které je uloženo, kde se tento program nachází. Dále je tu proměnná `$videoFile`, ve které se nachází umístění souboru. Poté jsou zde zbývající proměnné, které nás informují, jaké měřítko jsme vybrali a kam budeme překonvertované video ukládat.

Hlavním oříškem pro správnou práci programu je však nastavení. Nyní se podíváme na mé řešení. Já jsem se zaměřil na kompromis mezi rychlostí konverze a kvalitou výsledného videa. Tab. 4.1 uvádí informace o zvoleném nastavení.

Po vykonání konverze je nutné ještě výsledné video uložit, a to je definováno na konci příkazu. Celý tento příkaz je vykonán jako `exec()` příkaz, proto poté běží nezávisle na aplikaci. Pokud vše proběhne bez problému, tak se záznam o daném videu zanese do databáze. Zde budou k dispozici informace pro následné přehrávání či úpravy.

4.3 Databáze

Databáze je vytvořená pomocí programu MySQL Workbench, který jsem využíval pro její vytváření, správu i zálohu. Jak je vidět na obr. 4.2, tak ovládání je velmi intuitivní a rychle si na něj uživatel zvykne. Databáze samotná je velmi jednoduchá a do budoucnosti se dá rozšířit. Má v sobě uloženy informace o uživatelích a videích, které jsou na serveru.

Jak je vidět na schématu 4.3, databáze obsahuje dvě provázané tabulky, které v sobě nesou jednoznačné informace o uživateli nebo videoklipu. Většina práce s databází je čtení z ní, které má za následek zobrazení dané informace dle podmínky

Příkaz	Hodnota	Popis
-i	`$videoFile`	Nastavení vstupního videa.
-codec:v	libvpx	Nastavení kodeku pro video.
-quality	good	Nastavení rychlosti pro kodér VP8. Hodnoty jsou best, good, realtime.
-cpu-used	5	Nastavení kódování. Hodnoty jsou 0-5. Čím vyšší číslo, tím rychlejší kódování.
-b:v	500k	Nastavení bitové rychlosti pro video.
-qmin/-qmax	10/42	Jedná se o nastavení kvantizační hladiny pro minimální a maximální hodnotu.
-maxrate	500k	Příkaz pro omezení bitové rychlosti.
-bufsize	1000k	Identifikuje velikost bufferu.
-threads	4	Nastavení využití procesoru. Nastavuje se podle počtu jader CPU.
-vf scale	`$scale`	Definováno proměnnou, ve které je uloženo rozlišení videa.
-codec:a	libvorbis	Jedná se o zvukový kodér.
-b:a	128k	Bitová rychlost pro zvukovou stopu.

Tab. 4.1: Nastavení ffmpeg

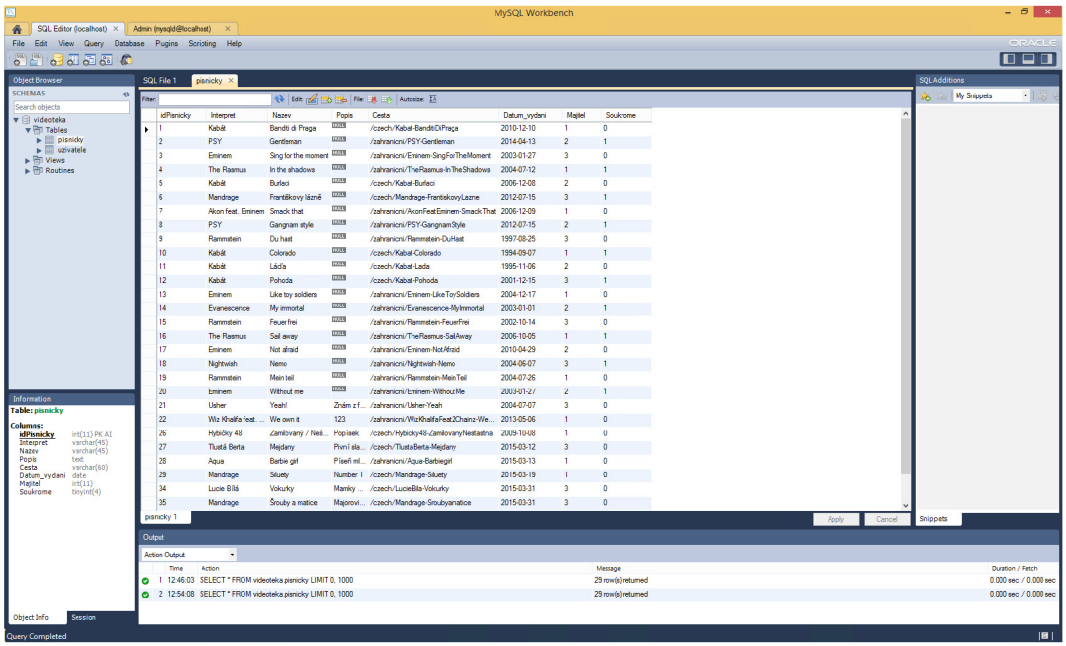
nebo porovnání informace s podmínkou. Samozřejmostí je i zápis do databáze, kdy se do ní zanáší nový uživatel či videoklip. Celá databáze běží na MySQL community serveru. Ten je spuštěn na počítači, proto je přítup k němu volán jako localhost. Je možné však využít poskytovatelů webhostingu, kteří již SQL databáze podporují a tím se dá vyřešit databázový i webový server zároveň. Na dalších řádcích je vypsáno několik zásadních příkazů a funkcí, které jsou v práci využity.

4.3.1 Vytvoření databáze

Databázi samotnou můžeme vytvořit mnoha způsoby. Nejčastěji k tomu využíváme programy, jako jsou phpMyAdmin nebo MySQL Workbench. Tyto programy za nás píší potřebné kódy a vytvoření databáze je i pro nezkušeného programátora velmi jenoduché. Následující kód nám ukazuje příklad vytvoření databáze. Je nutné kromě názvu zadat také kódování a jazyk, ve kterém budou jednotlivé tabulky vytvářeny.

```
CREATE DATABASE ‘nazev_databáze‘ CHARACTER SET utf8 COLLATE
utf8_czech_ci
```

Na jednom serveru můžeme provozovat několik databází a tím ušetřit využívané zdroje. Můžeme tak využít jeden databázový server pro několik serverů webových.



Obr. 4.2: Databáze v programu MySQL Workbench

4.3.2 Připojení databáze

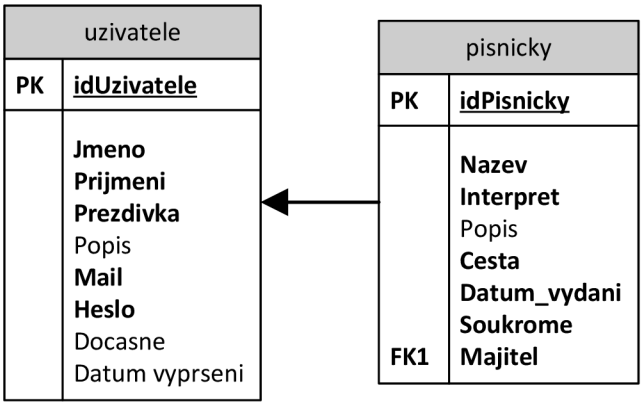
Pro připojení do databáze se využívá jednoduchého PHP skriptu, který je volán vždy, když je potřeba komunikace s databází. Je zde uloženo, kde se databáze nachází a jak se k ní připojit. Musí se zadat přihlašovací údaje a název databáze, ke které se připojujeme a se kterou chceme v budoucnu pracovat. Jelikož je funkce připojení volaná vždy při načítání ostatních MySQL PHP souborů je vytvořena separátně, aby se neopakoval zbytečně kód. Níže uvedený příkaz je nejdůležitější součást souboru. Díky němu je možné připojit se k serveru. Zde jsou uvedeny proměnné, ale dá se použít i přímé vyplnění. Pokud nedojde ke korektnímu připojení, tak se vypíše chybová hláška.

```
$link = mysql_connect(\$DBSERVER, \$DBUSER, \$DBPASS) or die("Could
  not connect: " . mysql_error());
```

I když se jedná o velmi jednoduchý skript, je zásadní pro celý projekt, proto je možné si jej prohlédnout v příloze A.2.

4.3.3 Vytvoření tabulky

Po vytvoření databáze samotné je nutné vytvořit tabulku, jinak by prázdná databáze byla k ničemu. Tabulky nám umožňují uchovávat data, která se dají třídit a zpracovávat různými způsoby. Aby bylo možné tabulku vytvořit, je nutné, aby se



Obr. 4.3: Schéma databáze videoteka

uživatel nejdříve připojil korektně k databázi, ve které chce danou tabulku vytvořit. Do té doby není možné tabulky vytvářet.

```
CREATE TABLE název_tabulky(sloupec 1 typ 1, ... , sloupec N typ N)
```

4.3.4 Vytvoření záznamu

Záznam do databáze se zanáší pomocí příkazu uvedeného níže. Pro záznam je nutné se připojit k databázi a následně se musí zadat správné údaje pro vložení do tabulky. Nutnost vložení je dána tím, jak byla tabulka vytvořena. Některé záznamy mohou být nepovinné, jako v mém případě popisky uživatele a videí, nebo samozapisovací. To zastupuje v mém případě primární klíč, což je idUzivatele respektive idPisnicky.

```
INSERT INTO jméno_tabulky VALUES (hodnota 1, ... , hodnota N)
```

4.3.5 Úprava záznamů

Úprava záznamu je důležitá pro uživatele hlavně z toho důvodu, že záznam nemusí mazat, ale stačí jej pouze upravit. Největší využití v projektu je úprava popisků ke klipům či jejich viditelnost pro ostatní uživatele. Té je v práci využito proto, aby sám uživatel mohl rozhodnout, zda chce, aby jeho video bylo viditelné či nikoliv.

```
UPDATE jméno_tabulky SET položka=hodnota, ... WHERE podmínka
```

4.3.6 Mazání záznamů

Mazání záznamů v databázi je důležité pro uvolnění zdrojů. Využívá se pro mazání často nevyužívaných záznamů nebo při přesunu do archivních tabulek, kde se

může uchovávat pouze zlomek informací. Projekt využívá mazání souborů pouze z iniciativy majitele záznamu. Ten jediný má právo mazat videa, která sám nahrál na server. Právě i v tomto případě se musí využít klauzule `WHERE`, podle které určíme záznam, jenž má být uživatelem smazán.

```
DELETE FROM jméno_tabulky WHERE podmínka
```

4.4 Funkce aplikace

Aplikace pro komplexní funkci využívá mnoho dalších funkcí, které se starají o bezproblémový běh celé aplikace. Jedná se o směs využití JavaScripu, PHP a databázových příkazů. U každé funkce je napsáno, který z jazyků využívá nebo jestli využívá jejich směs.

4.4.1 Přehrávání videí

Přehrávání videí je funkcí, o kterou se stará JavaScript samotný. Je většinou navázána na záznam z databáze, kdy se po kliknutí na zobrazený záznam načte přehrávač a se zpožděním začne přehrávat daný videoklip. Jelikož se jedná o hlavní funkci programu, tak se na ní můžete podívat v příloze A.1 Přehrávač se musí načítat z důvodu možnosti jeho absence (je umístěn v dynamické části programu). Následné zpoždění dodává čas k tomu, aby bylo vše korektně ukončeno a nedošlo k neočekávaným chybám, jako jsou pád přehrávače, načtení přehrávače do jiné části aplikace aj. Pozdější přehrávání daného klipu je určeno bufferováním souboru do paměti, aby se následně video přehrávalo bez "sekání". Tato skutečnost je závislá především na kvalitě videa, se kterou bylo na server nahráno.

4.4.2 Registrace uživatele

Zásadní část, která nešla opomenout je registrace uživatele. Jedná se o mix příkazů v PHP a MySQL jazyce. Po kliknutí na odkaz se otevře registrační formulář, který se odešle ke kontrole. Zde se zjistí, zda je již uživatelské jméno obsazeno. O zbylou kontrolu, zda je vše vyplněno, se stará JavaScript, obdobně jako u jiných formulářů, které se v aplikaci vyskytují. Poté se vytvoří záznam do tabulky uživatelů. O tuto skutečnost se stará MySQL.

Samozřejmostí je zahashování hesla uživatele. Je to z důvodu bezpečnosti, aby při vykradení databáze nebylo možné měnit záznamy. V práci využívám kombinaci hashování pomocí MD5 a SHA1. Není nutné využívat obě varianty zároveň. Pro zvýšení bezpečnosti je však vhodné, aby heslo bylo hashované několikrát. Pokud

heslo zahashujeme stejnou sekvencí třeba tisíckrát, tak je velmi nepravděpodobné, že konvenčí rozebrání hesel bude úspěšně realizované.

4.4.3 Logování uživatele

K logování uživatele dochází na úvodní stránce celé aplikace. Obdobně jako u registrace uživatele se i zde o řádný běh stará PHP a MySQL. Z formuláře se zjistí, jaké jméno a heslo zadal uživatel. Heslo se následně prožene hashovaní funkcí. Obě tyto informace zpracuje pro porovnání MySQL. Nejdříve hledá uživatele, který je v databázi jedinečný. Pokud jej najde, vybere se z tabulky heslo a porovná se se zadaným heslem. Pokud vše souhlasí, tak se nastaví superglobální proměnná login. Díky ní se poté zpřístupní některé další části aplikace, jako je vložení videa či zobrazení všech videí, které daný uživatel na server nahrál.

4.4.4 Vytváření záznamů a konverze klipů

Bezesporu hlavní část celé aplikace je právě práce s videem a především jeho konverze. Celý skript je psán v PHP jazyce. Pro konverzi je využito programu ffmpeg. Ten je uložen a spouštěn na straně serveru. Nejdříve se vyplní ve formuláři informace o videu a poté se vybere klip samotný. Tento celek se dále odešle na zpracování. Před samotným zpracováním videa se také řeší vytvoření náhledu souboru, resp. posteru. Je na uživateli, zda vloží svůj obrázek nebo ho vytvoří program sám. Potom se soubor videa analyzuje nástrojem ffprobe a s jeho pomocí se zjistí rozlišení, které je nutné pro další postup.

Podle poměru šířky a výšky se skript rozhodne, zda videa přetvoří na formát 4:3 nebo 16:9. V každé variantě se nachází čtyři vrstvy, přičemž každé odpovídá jiné rozlišení. Snažil jsem se vybrat podobné rozlišení pro jednotlivé poměry stran.

Dalším krokem je zpracování samotného videa. Pomocí příkazu uvedeném v části 4.2.2 se spustí program ffmpeg a začne s konverzí dle předem daných pravidel. Pokud vše proběhne v pořádku, tak se skript připojí k databázi a vytvoří záznam s danými informacemi, vše je následně ukončeno a přesměrováno na úvodní stránku.

4.4.5 Výběr dat z databáze

Výběr dat je možný pouze po korektním připojení k databázi. V práci je využito mnoho částí, které načítají tato data podle předem daných podmínek. Pro definici těchto podmínek se využívá klauzule WHERE. Jednotlivé podmínky se dají kombinovat a tím se dá dosáhnout velmi určitých výsledků.

Práce využívá nejvíce načítání podle žánru, který je definován v cestě k souboru. Je využito možnosti vyhledávání v textu, jak je uvedeno v 4.4.5. Další výběr dat

je podle data, uživatele nebo dle hledaného slova v případě vyhledávání konkrétního záznamu. Takováto data jsou prezentována uživateli, který definuje nevědomky podmínky podle daných odkazů.

PHP srovnání podle data

Příkaz pro srovnání podle data je velmi jednoduchý. Nejdříve se vyvolá soubor PHP, který má na starosti připojení k databázi a poté je vykonán příkaz `SELECT`, který vybere tabulku, a potom se načtou data do pole. Toto pole je srovnáno podle klíče, který jsme zadali. Na konci příkazu je způsob rovnání. Pro nás je to v obráceném pořadí, o to se stará příkaz `DESC`. Pokud je shodné datum, tak dojde na sekundární řazení podle interpreta. Poté se pomocí příkazu `while` projde celé pole a z něj si vybereme vše důležité, přičemž nás zajímá datum vydání, název klipu a interpret.

```
$query_sel = "SELECT * FROM ‘Pisnicky‘ ORDER BY ‘Datum_vydani‘ DESC,
  ‘Interpret‘ LIMIT 5";
```

PHP náhodné srovnání

Toto srovnání je velmi podobné jako srovnání podle data. Opět se připojíme k databázi pomocí externího souboru a následně vybereme náhodně pět prvků z pole. Daný náhodný výběr v sobě nikdy nebude obsahovat dvě stejné položky díky svému unikátnímu příkazu. Zde se vypisují jiné údaje, než u rovnání podle data. Jde konkrétně o vypsání interpreta a názvu klipu. Více není potřeba, protože to uživatele nebude zajímat.

```
$query_sel = "SELECT * FROM ‘Pisnicky‘ ORDER BY RAND() LIMIT 5";
```

PHP srovnání podle typu skladby

Poslední typ zobrazení, který se velmi často využívá je zobrazení určitého typu skladeb. V tomto případě jsou rozděleny na české a zahraniční klipy. Toho se využívá pro následné zobrazení.

```
$query_sel = "SELECT * FROM ‘Pisnicky‘ WHERE cesta LIKE ’%zahranic-
ni%’ and  Soukrome = 0 ORDER BY ‘Datum_vydani‘ DESC, ‘Interpret‘
LIMIT 8";
```

Na příkladu je hlavní příkaz pro zobrazení zahraničních klipů. Jsou zde uvedeny dvě podmínky, které musí být splněny. Zaprvé musí cesta obsahovat slovo zahraniční, navíc musí být video veřejné. Dodatečné nastavení příkazu je řazení, které je primárně podle data vložení a sekundárně podle interpreta.

4.4.6 Vyhledávání

Vyhledávání jako takové je řešeno dvěma způsoby. Prvním z nich je možnost vyhledávání na úvodní obrazovce, kde je umístěna atribut `<input>`, kam se po zadání znaku zobrazí nabídka odpovídajících výsledků. Takto se situace opakuje do té doby, dokud jsou možnosti rozhodnutí. Seznam, ve kterém se hledá, je seznam interpretů. Ten se získal projítím databáze a zaznamenáním interpreta do pole prvků. Toto pole pak bylo předáno JavaScriptové aplikaci, která nám pak nabízí nabídky.

K dispozici je rovněž klasické vyhledávání, které po zadání hledaného slova projde databázi a najde odpovídající výsledky. Tato varianta podporuje i vyhledávání v názvech klipů. Záleží tak na uživateli, kterou ze zmíněných možností zvolí.

4.4.7 Obnova hesla

Ne každý uživatel si dokáže po době nevyužívání svého účtu vzpomenout na své heslo. Proto i v tomto projektu nechybí možnost jeho obnovy. Celý proces se skládá z několika částí. Nejdříve po kliknutí na odkaz obnovy hesla se objeví vstupní pole, do kterého uživatel vypíše svojí e-mailovou adresu. O kontrolu správnosti adresy se stará PHP soubor, který prohledává databázi. Pokud je zadaná adresa platná, vygeneruje se náhodné číslo, které se generuje ve smyčce a následně sčítá s dalším náhodným číslem. Výsledné číslo se následně zahashuje a uloží do databáze s aktuálním datem k danému uživateli. Dále je na registrační e-mail zaslán odkaz pro obnovu samotnou.

Další částí je obnova samotná. V odkazu je zakódované dočasné heslo, kterým se určuje uživatel, který žádal o proces obnovy hesla. Program detekuje obnovovací adresu a nabídne uživateli možnost vložení nového hesla. Pokud projde heslo kontrolou, tak nastane další část kontroly. Tou se rozumí kontrola data. V aplikaci jsem omezil možnost změny do jednoho měsíce od inicializace příkazu. Pokud se tato doba vyčerpá, tak musí uživatel zažádat o změnu hesla znovu.

Po celou dobu obnovy hesla je staré heslo stále platné. Pokud proces dojde až do finální fáze, tak se uvolní databázový prostor pro další využití.

4.4.8 Odlogování uživatele

Poslední funkcí, která je neméně důležitá, je odhlášení uživatele. Stará se o ní pouze PHP skript, který ukončuje session a všechny superglobální proměnné, které byly během doby, kdy byl uživatel přihlášen vytvořeny. Dochází díky tomu k uvolnění prostředků pro další akce a hlavně strojového času serveru samotného. Jednoduchý skript je uveden v příloze A.3.

5 MĚŘENÍ RYCHLOSTI KONVERZE VIDEÍ

Zásadním přínosem práce je zjištění možnosti aplikace samotné. Za tímto účelem byla přidána funkce, která měří rychlost konvertování videa. Výsledný čas každé konverze je ukládán spolu s názvem klipu do textového souboru. Odtud je poté možné vyčíst částečně výsledky.

Z tohoto řešení těží první typ měření. Další měření bylo provedeno za účelem zjistit, jak se bude prodlužovat, nebo zkracovat doba, pokud budou videa zpracovávána paralelně či sériově. Pro paralelní typ měření jsem měl k dispozici deset stanic, které začaly konvertovat stejné video ve stejnou dobu. Sériové využití jsem zajistil pomocí zřetězení jednoho až deseti příkazů.

Veškeré měření probíhalo na virtuálním stroji, na kterém byl nainstalován 64-bitový operační systém Windows 8.1 Pro. O výkon se staral čtyřjádrový procesor Intel Xeon E5520 s taktovací frekvencí 2,27 GHz na jedno jádro a podporou HyperThreadingu (osm virtuálních jader). K dispozici bylo celkem 8 GB operační paměti, což zaručovalo dostatečný výkon pro běh webového serveru.

O jednotlivých měřeních se dozvíte v následujících kapitolách.

5.1 Závislost typu souboru na době konverze

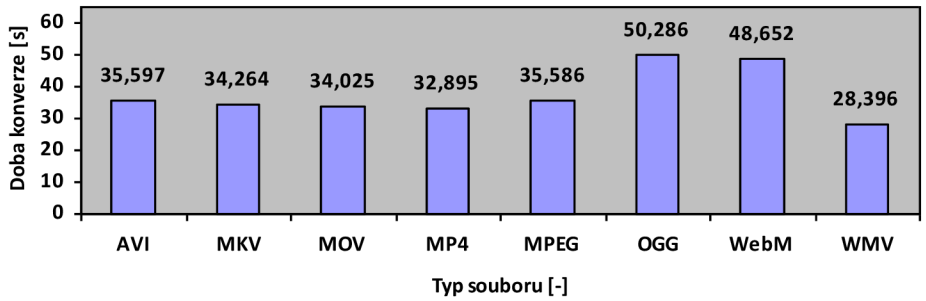
Pro možnosti toho měření jsem vytvořil osm souborů, které si byly velmi podobné ve svých parametrech. Zaměřil jsem se hlavně na kontrolu výstupních souborů tak, aby měly shodné snímkování a hlavně bitový tok. Jako zdroj byl vybrán soubor, který měl parametry krátkého a velmi kvalitního videa. Podrobnosti o zdrojovém souboru jsou vyneseny v tab. 5.1. Pro každý soubor jsem musel najít správný kodér a nastavení pro ffmpeg, abych zajistil vzájemnou porovnatelnost jednotlivých klipů.

Informace	Hodnota
Video kodek	h264
Rozlišení	1920:1080
Poměr stran	16:9
Bitový tok	8138 kb/s
Počet snímků za vteřinu	23,98 fps
Zvukový kodek	ac3
Vzorkovací frekvence	640 kb/s

Tab. 5.1: Informace o zdrojovém souboru

5.1.1 Postup měření

Po připravení osmi testovacích souborů bylo nutné změřit jejich konverzi. Pro vynesení do grafu stačilo v aplikaci vkládat jednotlivá videa a následně si zobrazit jednotlivé časy konverze. To umožňovala vestavěná funkce PHP pro měření, resp. zjištění času. Jednotlivé časy konvertování jsou zaneseny přímo v grafu 5.1.



Obr. 5.1: Závislost doby konverze na typu souboru

5.1.2 Shrnutí

Podle měření, které se opakovalo desetkrát po sobě a zanesení odchylky vyšlo, že nejrychleji bylo převedeno video ve formátu `.wmv`. Nejpomaleji se převáděla videa ve formátu `.ogg`. To bych přisuzoval tomu, že jak `.ogg`, tak i `.webm` využívají ke konvertování stejné kodeky, proto je jejich doba pro překonvertování nejdelší. Naopak je tomu u sady kodeků `MPEG`, které patří mezi rychlejší.

5.2 Závislost mnohonásobné konverze na její době

Toto měření jsem rozdělil na sériové a paralelní proto, aby bylo vidět, jak se projevuje zpracování jednotlivých požadavků. V reálném případě může dojít k tomu, že část požadavků je odbavována sériově, jako v případě, kdy jeden uživatel nahrává více videí, nebo paralelně, kdy je připojeno k aplikaci více uživatelů ve stejnou dobu a využívají možnosti konverze videa.

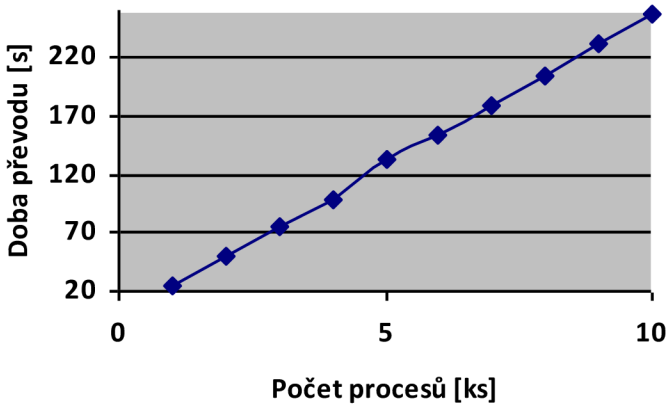
5.2.1 Sériové zpracovávání

Jak již bylo řečeno, tento typ přístupu je omezen na jednoho uživatele. Ten může vkládat více videí za sebou a tím využít strojový čas prostředků serveru.

První, čeho si správce serveru všimne v tomto případě, je využití procesoru. Při zadání požadavku aplikace ffmpeg využije všechen volný strojový čas procesoru. Díky tomu vyhoví uživateli a zpracuje daný požadavek v co nejkratší době. Po splnění jednoho požadavku teprve začne zpracovávat následující skript nebo video.

Postup měření

Pro toto měření jsem vytvořil skript s jednoduchou smyčkou, kdy po každém průchodu bylo konvertováno jedno video. Měnil jsem pouze velikost smyčky, abych zjistil, jaký vliv má větší počet souborů, resp. požadavků, které budou zpracovávány. Měřil se tudíž čas konverze jednoho až deseti požadavků v sériovém sledu. Hodnoty z měření je možné si prohlédnout v tab.B.1.



Obr. 5.2: Závislost doby konverze na sériovém zpracovávání požadavků

Výsledky

Jelikož se jedná o sériové požadavky, tak se dalo předpokládat, že doba se bude konstantně zvětšovat o dobu zpracování jednoho požadavku. To se i měřením potvrdilo. Graf 5.2 má lineární závislost. Jemné niance jsou způsobeny řízením programu

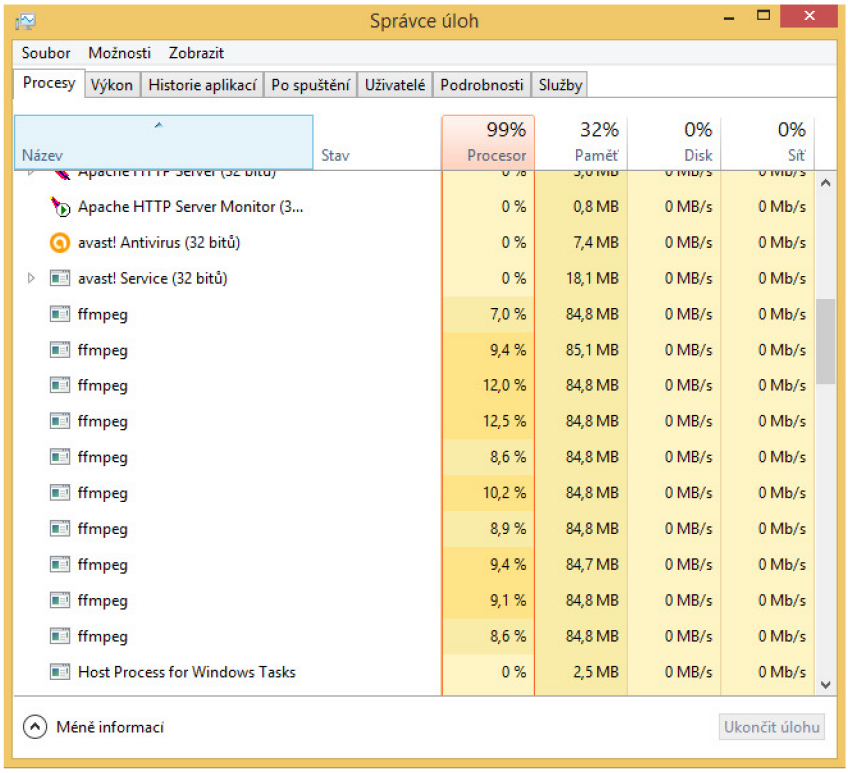
ffmpeg. Jednotlivé rozdíly mezi akcemi jsou téměř 26 sekund v průměru, což je jen o málo více, než je trvání převodu jednoho videa.

5.2.2 Paralelní zpracovávání

Paralelní zpracování je v jazyce PHP velmi obtížné realizovat, protože vykonává své příkazy sériově. Proto jsem pro toto měření využil služeb uživatelů, kteří zpracovávali jedno video ve stejnou dobu. Měnil jsem počet uživatelů a zaznamenával čas, který byl potřeba pro konvertování.

Postup měření

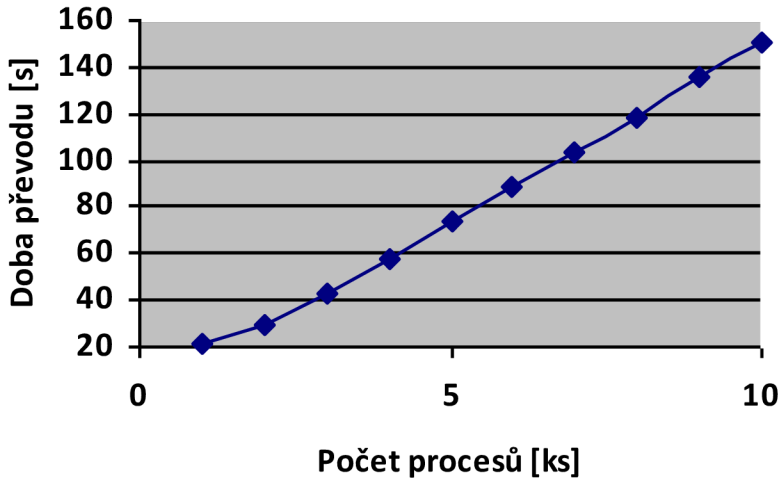
Pro měření jsem zajistil deset pracovních stanic, které jsem připojil k počítači na kterém se provádělo měření. Zvukovým znamením se startovalo konvertování jednotlivých uživatelů. Nejdříve jednoho, druhého atd. Pro korektnost se toto měření provádělo desetkrát po sobě. Hodnoty, které se během měření změřily jsou uvedeny v tab.B.2. Pro zajímavost uvádím i obr. 5.3, který zobrazuje aktuální využití procesoru serveru. Jak se dalo předpokládat, tak každý spuštěný proces se podělil s jiným stejným procesem o přibližně stejný strojový čas.



Obr. 5.3: Využití procesoru během paralelního měření

Výsledky

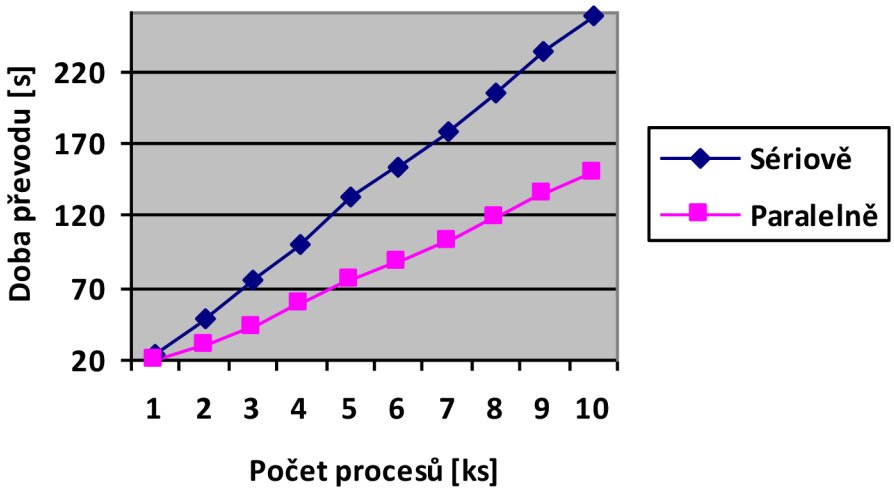
Při prozkoumání naměřených hodnot a vykresleného grafu 5.4 lze zjistit, že při paralelním zpracovávání dochází také k téměř lineární závislosti času na počtu zpracovávaných požadavků. Pokud vynecháme iteraci jednoho prvku, kde je měření nerelevantní z důvodu absence paralelního zpracování, je delta mezi jednotlivými přidanými procesy 15 sekund, což je přibližně o pět sekund rychlejší zpracování, než zpracování jednoho samotného souboru.



Obr. 5.4: Závislost doby konverze na paralelním zpracovávání požadavků

5.2.3 Shrnutí

Při pohledu na graf 5.5 je patrné, že paralelní zpracování příkazů na konvertování videí je mnohem výhodnější než seriové zpracování. Obecně se dá říci, že čím více požadavků, tím výhodnější paralelní zpracování je. V mém případě je při deseti požadavcích rozdíl ohromných 108,2 sekundy, což odopovídá zpracování dalších sedmi požadavků, abychom se dostali na stejnou dobu, co trvá deset sériově řešených příkazů. Díky struktuře PHP jazyka je ale paralelní zpracování dosti obížně realizovatelné. Aplikace sama o sobě pracuje jako sériové zpracování, ale pokud několik uživatelů ve stejnou dobu bude využívat službu nahrávání, tak se dostaví i aplikace paralelního zpracování.



Obr. 5.5: Závislost doby konverze na sériovém a paralelním zpracovávání požadavků

6 ZÁVĚR

Cílem práce bylo vytvoření aplikace, která by umožňovala Audio/Video streaming. Během řešení jsem vytvořil program, který zahrnoval správu jak uživatelů, tak videí. Všechna videa byla konvertována a nahrávána na server.

Zásluhou spojení MySQL, PHP a JavaScriptu bylo možné uskutečnit klientskou část, avšak bylo nutné zprovoznit lokální server pro testování. Tohoto úkonu se ujal server Apache, který je nejrozšířenějším open source programem po celém Internetu doplněný o podporu PHP. Tento celek pak doplnil MySQL community server pro uchovávání databáze a program ffmpeg, který se staral o konvertování videí.

Nakonec se zjistila schopnost serveru, jak rychle dokáže zpracovat jednotlivé typy formátů. A dále se zkoumalo, jak se projevuje typ zpracování požadavků na rychosti konvertování videí. Z měření pro jednotlivé soubory bylo nejrychleji převedeno video ve formátu `.wmv`. Nejdéle trval převod z formátu `.ogg`. Při porovnávání jednotlivých typů zpracování se došlo k závěru, že paralelní řešení je mnohem výhodnější než sériové. Rozdíl mezi deseti příkazy řešenými sériovou a paralelní cestou byl více než sto sekund, což umožňovalo paralelní variantě zpracovat více požadavků za stejný časový úsek.

Práce pouze nastínila jedno z mnoha východisek, jak by zadaný problém mohl být řešen.

LITERATURA

[1] LUBBERS, Peter, Brian ALBERS a Frank SALIM. *HTML5: programujeme moderní webové aplikace*. Vyd. 1. Brno: Computer Press, 2011, 304 s. ISBN 978-80-251-3539-6.

[2] DOMES, Martin. *Tvorba internetových stránek pomocí HTML, CSS a JavaScriptu*. Vyd. 1. Kralice na Hané: Computer Media, 2005, 324 s. ISBN 80-86686-39-6.

[3] POKORNÝ, Martin. *PHP nejen pro začátečníky*. Vyd. 1. Kralice na Hané: Computer Media, 2005, 228 s. ISBN 80-86686-38-8.

[4] PONKRÁC, Miloslav. *PHP a MySQL: bez předchozích znalostí : [průvodce pro samouky]*. Vyd. 1. Brno: Computer Press, 2007, 221 s. ISBN 978-80-251-1758-3.

[5] DUBOIS, Paul. *MySQL profesionálně: komplexní průvodce použitím, programováním a správou MySQL*. Vyd. 1. Brno: Mobil Media, 2003, 1071 s. ISBN 80-86593-41-x.

[6] LACKO, Ľuboslav. *PHP 5 a MySQL 5: hotová řešení* . Vyd. 1. Brno: Computer Press, 2007, 320 s. ISBN 978-80-251-1695-1.

SEZNAM SYMBOLŮ, VELIČIN A ZKRATEK

HTML	Označení pro značkovací jazyk, ve kterém se píší hypertextové dokumenty – HyperText Markup Language.
WWW	World Wide Web - celosvětová sít pro aplikace internetového protokolu
tag	Označení pro značku v HTML jazyce.
WWE	World Wrestling Entertainment - zábavní společnost provozující wrestling v USA
ID	identifikátor púro označení prvku v HTML stránce
GPL	General Public License - licence pro volně šiřitelné programy
SQL	Structured Query Language - strukturovací dotazovací jazyk
PHP	PHP: Hypertext Preprocessor - skriptovací programovací jazyk určený pro vytváření webových stránek
API	zkratka pro označení již vytvořené aplikace
ECMA	European Computer Manufacturers Association
ISO	International Organization for Standardization - mezinárodní organizace pro normalizaci
HTTP	Hypertext Transfer Protokol - protokol, který zajišťuje výměnu dokumentů ve formátu HTML
CPU	Central processor unit - označení pro porcesorovou jednotku

SEZNAM PŘÍLOH

A JEDNOTLIVÉ ČÁSTI KÓDU APLIKACE

Zde jsou uvedeny hlavní části programu, které jsou v práci nejdůležitější. Hlavní část, bez které by program nefungoval, je JavaScriptová funkce pro přehrávání videa. Neméně důležitou se stala funkce připojení k databázi, bez které by nebylo možné s databází vůbec pracovat. Pro korektní ukončení všech relací byla potřebná i funkce odlogování uživatele. O jednotlivých částech je napsáno níže.

A.1 JavaScript

JavaScript na začátku načte proměnnou přehrávač. Poté je funkční pouze při užití funkce `nacti()`. Do té vstupuje cesta ke klipu, interpret a název klipu. Je tomu tak proto, že cestu využijeme k lokalizaci videa a zbytek je využívám pro informační návěstí umístěné nad prvkem `<video>`.

```
<script>
window.onload = function (){
  var prehravac = document.getElementById("preravac");
  txtStatus.innerHTML = "<i><------- <b>HTML5 prehravac videa!
    </b>------><i>";
}

function nacti(cesta, interpret, nazev){
  var cesta = cesta;
  var interpret = interpret;
  var nazev= nazev;
  txtStatus.innerHTML = "<i><------- [<b>" + interpret + "</b>]" +
    nazev + " ------><i>";
  prehravac.src= "video/" + cesta;
  prehravac.play();
};
</script>
```

A.2 Připojení k databázi

Připojení k databázi si zasloužilo svůj vlastní soubor PHP. Tohoto řešení se využívá, aby nebylo nutné neustále opakovat kód, což by vedlo k neefektivnosti kódu. Na začátku jsou uloženy proměnné, které definují adresu serveru, uživatele, pod kterým

se připojuje k databázi, heslo uživatele a název databáze. Příkaz `mysql_connect()` připojí aplikaci k databázovému serveru. Pokud se tak nestane, vyskočí chyba. Poslední příkaz, kterého je využito je `mysql_select_db()`, která se stará o připojení ke konkrétní databázi. Na závěr se pouze nastaví kódování záznamů, které se budou využívat.

```
<?php
$DBSERVER = 'localhost';  //nastavení serveru, kde běží databáze
$DBUSER = 'root';         //login na server
$DBPASS = '';             //heslo, pro daný login

$DB = 'Videoteka';        //databáze, se kterou budu pracovat

//navázání spojení pomoci mysql_connect
$link = mysql_connect($DBSERVER, $DBUSER, $DBPASS) or die("Could
not connect: " . mysql_error());
//vybrání databáze
mysql_select_db($DB, $link) or die ('Can\'t use Videoteka : '
.mysql_error());
//řekněme MySQL, že všechná data jsou ve formatu UTF-8
mysql_query("SET NAMES UTF8");
?>
```

A.3 Odlogování uživatele

Neméně důležitou částí je odlogování uživatele. Jde především o uvolnění zdrojů, které jsou velmi cenné. Při každé akci uživatele se volá funkce `session_start()`, kde jsou uloženy superglobální proměnné. Pokud se uživatel odloguje, tak právě funkce `session_destroy()` smaže všechny proměnné vytvořené v rámci session.

```
<?php
session_start();/* Zapneme session */
session_unset();
session_destroy();/* Smažeme všechna session */
echo "Odhlášeno";
header("Location: ../index.php"); /* Přesmeruje na přihlašovací
stránku */
?>
```

B MĚŘENÍ

V této části příloh jsou vyobrazeny tabulky měření. Jednotlivá měření probíhala desetkrát pro lepší validitu výsledku a jejich hodnoty si můžete prohlédnout níže.

B.1 Sériové měření

Jedná se o měření sériových požadavků, kdy jednotlivé příkazy následovaly jeden za druhým.

Počet měření	1	2	3	4	5	6	7	8	9	10	Půměr
Počet příkazů	Čas [s]										
1	24,32	24,31	24,33	24,32	24,32	24,33	24,32	24,33	24,32	24,34	24,324
2	49,62	49,65	49,63	49,64	49,63	49,63	49,62	49,63	49,62	49,51	49,618
3	74,82	74,83	74,82	74,81	74,83	74,82	74,82	74,83	74,83	74,82	74,823
4	99,54	99,56	99,54	99,56	99,55	99,56	99,54	99,56	99,53	99,54	99,548
5	132,24	132,22	132,23	132,24	132,22	132,23	132,22	132,24	132,24	132,22	132,23
6	154,22	154,21	154,22	154,22	154,23	154,23	154,22	154,21	154,21	154,22	154,219
7	178,92	178,91	178,92	178,92	178,93	178,91	178,91	178,92	178,91	178,92	178,917
8	204,11	204,1	204,12	204,11	204,11	204,12	204,12	204,11	204,1	204,12	204,112
9	232,35	232,34	232,34	232,35	232,34	232,35	232,33	232,34	232,35	232,34	232,343
10	258,46	258,45	258,45	258,46	258,47	258,45	258,46	258,45	258,46	258,45	258,456

Tab. B.1: Měření sériových požadavků

B.2 Paralelní měření

Při měření paralelních požadavků jsme simulovali provoz na serveru pomocí deseti pracovních stanic, které začaly konvertovat video ve stejný čas. Nejdříve prováděla příkaz jedna stanice, pak dvě, tři atd. Jednotlivá měření probíhala desetkrát, aby byla lepší validita výsledku.

Počet měření	1	2	3	4	5	6	7	8	9	10	Půměr
Počet příkazů	Čas [s]										
1	20,72	20,81	20,73	20,78	20,77	20,75	20,75	20,71	20,78	20,74	20,754
2	29,71	29,72	29,61	29,53	29,62	29,58	29,61	29,71	29,65	29,64	29,638
3	43,21	43,30	43,23	43,22	43,23	43,21	43,25	43,20	43,19	43,23	43,227
4	58,26	58,30	58,23	58,21	58,21	58,24	58,23	58,24	58,31	58,24	58,247
5	74,56	74,51	74,51	74,52	74,52	74,52	74,50	74,48	74,43	74,45	74,50
6	88,69	88,69	88,74	88,69	88,70	88,72	88,74	88,71	88,70	88,72	88,71
7	102,83	102,79	102,81	102,8	102,82	102,84	102,81	102,84	102,81	102,81	102,816
8	118,43	118,43	118,41	118,42	118,44	118,43	118,44	118,41	118,42	118,43	118,426
9	135,19	135,2	135,18	135,17	135,14	135,18	135,17	135,18	135,16	135,16	135,173
10	150,21	149,99	150,32	150,35	150,28	150,29	150,29	150,27	150,30	150,23	150,253

Tab. B.2: Měření paralelních požadavků

C OBSAH PŘILOŽENÉHO CD

Na přiloženém mediálním disku je možné nalézt digitální podobu této práce a její aktuální provedení. Jedná se především o klientskou aplikaci, která po správném nainstalování přiložených souborů dokáže pracovat jako videotéka. Celá aplikace je umístěna ve složce HTML a je možné ji spustit skrze webový server Apache jako soubor `index.php`. Tato složka obsahuje i kódy PHP, které umožňují práci s databází. Výsledný grafický layout je umístěn v souboru `Kaskada.css`. Ten je také oddělem ve složce CSS.

Dále jsou zde k nalezení instalační soubory, které jsou nutné pro správný běh aplikace. Pro jejich snadnější instalaci obsahuje médium i textový soubor `install.txt`, ve kterém je uveden postup pro správnou instalaci. Všechny tyto soubory jsou umístěny ve složce INSTALACE.