Sem vložte zadání Vaší práce.

České vysoké učení technické v Praze
Fakulta informačních technologií
Katedra softwarového inženýrství

Bakalářská práce

Automatizovaný vyhledávač pirátských kopií

David Vondraš

Vedoucí práce: Ing. Robert Pergl, Ph.D.

5. května 2015

Poděkování

Rád bych poděkoval vedoucímu práce Ing. Robertu Perglovi, Ph.D. za poskytnuté rady při psaní mé bakalářské práce.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval(a) samostatně a že jsem uvedl(a) veškeré použité informační zdroje v souladu s Metodickým pokynem o etické přípravě vysokoškolských závěrečných prací.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona, ve znění pozdějších předpisů. V souladu s ust. § 46 odst. 6 tohoto zákona tímto uděluji nevýhradní oprávnění (licenci) k užití této mojí práce, a to včetně všech počítačových programů, jež jsou její součástí či přílohou, a veškeré jejich dokumentace (dále souhrnně jen „Dílo“), a to všem osobám, které si přejí Dílo užít. Tyto osoby jsou oprávněny Dílo užít jakýmkoli způsobem, který nesnižuje hodnotu Díla, a za jakýmkoli účelem (včetně užití k výdělečným účelům). Toto oprávnění je časově, teritoriálně i množstevně neomezené. Každá osoba, která využije výše uvedenou licenci, se však zavazuje udělit ke každému dílu, které vznikne (byť jen zčásti) na základě Díla, úpravou Díla, spojením Díla s jiným dílem, zařazením Díla do díla souborného či zpracováním Díla (včetně překladu), licenci alespoň ve výše uvedeném rozsahu a zároveň zpřístupnit zdrojový kód takového díla alespoň srovnatelným způsobem a ve srovnatelném rozsahu, jako je zpřístupněn zdrojový kód Díla.

V Praze dne 5. května 2015 .....

Odkaz na tuto práci

Vondraš, David. *Automatizovaný vyhledávač pirátských kopií.* Bakalářská práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2015.

Abstrakt

Tato bakalářská práce si klade za cíl navrhnout a implementovat aplikaci sloužící na vyhledávání nelegálních kopií děl na internetu. Součástí práce je zjištění konkrétních požadavků zadavatele, dle kterých je proveden návrh a implementace aplikace. Hlavním cílem práce je navrhnout vhodnou architekturu aplikace a dále vybrat vhodné technologie, pomocí kterých bude aplikace realizována.

Klíčová slova pirástká kopie, sdílení souborů, REST, webová aplikace, responzivní web design

Abstract

This thesis aims to design and implement an application used to search illegal copies of works on the Internet. Part of this work is to determine the specific requirements of the client, according to which is made the design and implementation of the applications. The main objective is to design a suitable architecture of the application and select the appropriate technology by which the application will be implemented.

Keywords pirate copy, file sharing, REST, web application, responsive web design

Obsah

Seznam obrázků

Úvod

Na internetu se objevuje stále více nelegálních kopií, které jsou, na rozdíl od originálních distribucí, volně ke stažení. Tím vznikají majitelům autorských práv velké finanční ztráty. Ti se proto snaží proti pirátům bránit a nelegální kopie svých děl z internetových serverů odstraňovat. To je ale vzhledem k velkému počtu serverů zaměřených na sdílení souborů velice časově náročné. Aby bylo možné pirátské kopie odhalit, je potřeba pravidelně monitorovat vyhledávací servery. To bez vyhledávacího softwaru vyžaduje zdlouhavou manuální práci při zadávání klíčových slov do dialogů vyhledávacích serverů a procházení výsledků. Vzhledem k této náročnosti není reálný pravidelný monitoring, a tak se stává, že se nahlášený a odstraněný soubor velmi brzy na serveru opět objeví. Přestože těchto souborů na internetu stále přibývá, existuje velice málo možností jak tomu zabránit.

Tato práce si klade za cíl navrhnout a naimplementovat zkušební verzi aplikace, která celý proces vyhledávání nelegálních kopií co nejvíce zrychlí a zautomatizuje, a aby bylo možné nelegální kopie souborů efektivněji odhalovat a odstraňovat se serverů při eliminaci časově náročné manuální práce.

V první části se budu zabývat rešerší podobných řešení, návrhem aplikace a výběrem správných technologií, ve druhé realizací a v poslední části testováním a zkušebním nasazením.

Kapitola 1

Upřesnění zadání

Vytvořit systém, který by pomohl při odstraňování nelegálních kopií z internetu, byl nápad pana nakladatele Patrika Jandy z nakladatelství Alfa. Sešel jsem se s panem nakladatelem a projednali jsme podrobnosti zadání práce. Původní vize byla, že aplikace bude sloužit pouze pro vyhledávání nelegálních kopií knih. Zadání bylo později rozšířeno na hledání všech typů souborů. Vzhledem k veliké oblíbenosti mobilních zařízení bylo požadavkem pana nakladatele pohodlné použití aplikace na mobilním zařízení. Aby nebylo nutné výsledky vyhledávání v aplikaci neustále kontrolovat, vznikl požadavek na odesílání výsledku hledání emailem.

Kapitola 2

Cíle práce

Cílem této bakalářské práce je vytvořit prototypovou implementaci aplikace, která bude sloužit k efektivnímu vyhledávání pirátských kopií děl a pomůže při odstraňování těchto děl ze serverů, které jsou zaměřeny na sdílení souborů. Hlavním cílem práce je navrhnout vhodnou architekturu aplikace a dále vybrat vhodné technologie, pomocí kterých bude aplikace realizována.

Kapitola 3

Rešerše

Důvodem pro vznik této bakalářské práce byl fakt, že podobných systémů existuje velice málo. V České republice existuje pouze jedna služba, která se zabývá vyhledáváním pirátských kopií knih na internetu s názvem eBook-Service. Tato služba kromě automatického prohledávání serverů na sdílení souborů umožňuje prohledáváni českého internetu a evidenci stránek, které obsahují nelegální kopie děl, nebo právní poradenství a jednání se správci serverů v případě problematického stažení nelegálních kopií[2]. Uživatel za tyto služby zaplatí od 300 do 1000 korun za jedno dílo měsíčně podle toho, jak často probíhá vyhledávání. To znamená pro nakladatelství značnou finanční náročnost.
Podobná služba, která se zaměřuje také na vyhledávání e-knih, je Digimarc Guardian. Tato služba vyhledává podle názvu knihy, autora, formátu a jazyka, ve kterém je kniha napsána. Každý podezřelý soubor je dvakrát zkontrolován a pak je nahlášen serveru, který by ho měl odstranit.

Kapitola 4

Funkční a nefunkční požadavky

4.1 Funkční

F1. Automatické vyhledávání v zadaných časových intervalech

- Aplikace bude v zadaných časových intervalech spouštět vyhledávání na definovaných serverech.

F2. Automatické odesílaní informace o nalezení pirátské kopie souboru emailem

- Aplikace bude obsahovat funkci, která umožní automatické odeslání informace o nalezených souborech na email uživatele.

F3. Možnost vytvoření uživatelského účtu

- Uživatelský účet bude tvořen emailem a heslem, které bude možné změnit v nastavení. Součástí nastavení bude i emailová adresa, na kterou se budou odesílat výsledky vyhledávání.

F4. Přidávání a odstraňování klíčových slov pro vyhledávání

F5. Zobrazení výsledků vyhledávání

- Aplikace umožní zadat vyhledávací kritéria pro zobrazení výsledků.

F6. Změna nastavení frekvence vyhledávání

- Uživatel bude mít možnost si v nastavení aplikace vybrat časový interval, po jehož uplynutí se bude opakovaně spouštět vyhledávání.

4.2 Nefunkční

N1. Jednoduché ovládání

- Aplikace bude ovládána pomocí intuitivního uživatelského prostředí.

N2. Přizpůsobení uživatelského rozhraní různým velikostem zařízení

- Aplikace musí automaticky přizpůsobovat uživatelské rozhraní podle velikosti zobrazovacího zařízení, aby bylo možné pohodlně procházet výsledky vyhledávání na jakémkoli zařízení s přístupem na internet.

N3. Rozšiřitelnost a modifikovatelnost

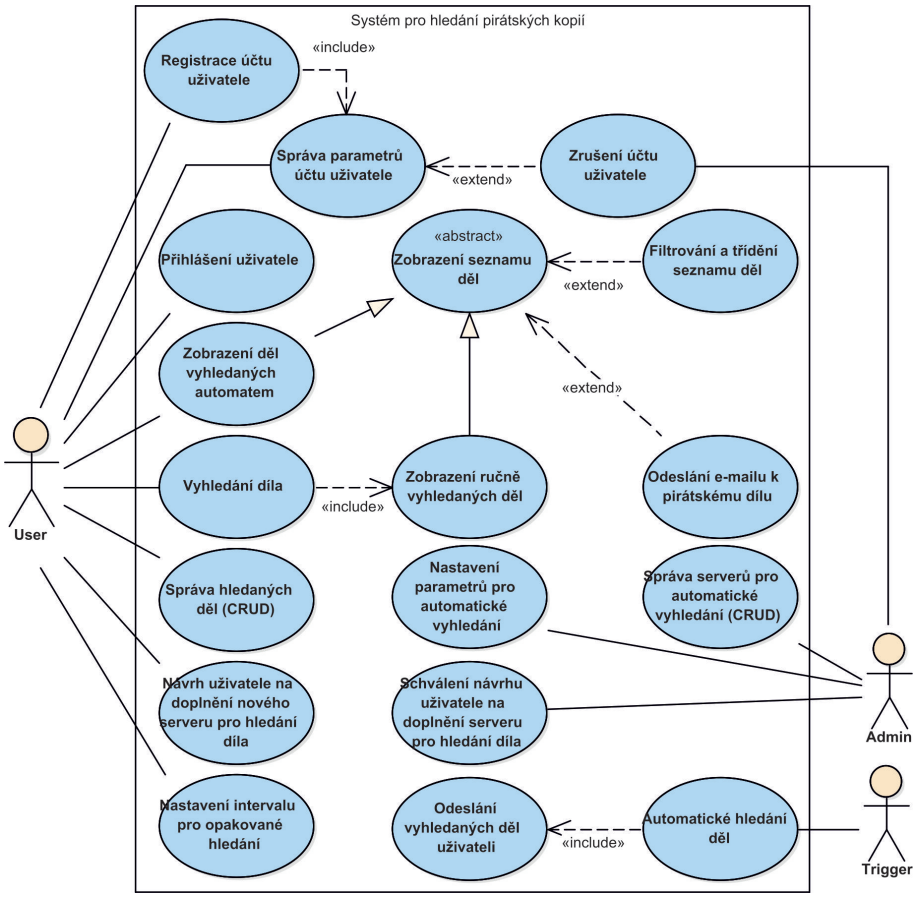
- Aplikace musí umožňovat reakci na změnu struktury prohledávaného serveru a umožňovat rozšiřování vyhledávání o další servery.

N4. Přenositelnost na různé platformy

- Aplikace bude umožňovat spuštění na různých mobilních i desktopových systémech.

Kapitola 5

Případy užití



Obrázek 5.1: Diagram případů užití

Registrace účtu uživatele
Aby uživatel mohl aplikaci využívat, musí si vytvořit uživatelský účet. Registrace vyžaduje jméno, heslo a email nového uživatele. Po úspěšném zaregistrování je účet okamžitě aktivován a lze s nim pracovat. Pokud aktivace není úspěšná, je uživateli zobrazena chybová hláška.

Přihlášení uživatele
Pokud má uživatel vytvořen účet v aplikace, musí se před použitím aplikace přihlásit. Uživatel zadává jméno a heslo. Pokud je přihlášení úspěšné, je uživateli povolen přístup do aplikace. Pokud je přihlášení neplatné, je uživateli zobrazena chybová hláška.

Správa parametrů účtu uživatele
Funkce umožňuje uživateli změnu emailové adresy a hesla, pod kterým se do aplikace přihlašuje.

Nastavení intervalu pro opakované hledání
Funkce umožňuje uživateli změnit časový interval, po kterém se spouští pravidelné vyhledávání souborů na serverech. Uživatel si vybírá z předem definovaného seznamu intervalů.

Správa hledaných děl
Funkce zobrazí seznam zadaných souborů, které si uživatel přeje vyhledávat, a zpřístupní mu možnost přidávat a mazat vyhledávané názvy souborů. Každý uživatel může spravovat pouze soubory přiřazené ke svému účtu.

Návrh uživatele na doplnění nového serveru pro vyhledávání díla
Funkce dává uživateli možnost odeslat zprávu správci aplikace, obsahující název serveru, který si přeje přidat do vyhledávání.

Zobrazení děl vyhledaných automatem
Uživatel má možnost zobrazit si seznam souborů, nalezených automaticky spouštěným vyhledávacím systémem. Uživatel si může před odesláním příkazu na stažení a zobrazení výsledků hledání nastavit filtr. Filtr se nastavuje výběrem hodnot ze seznamu názvů souborů, seznamu serverů, typu souborů a příznaku, zda již uživatel soubory zobrazil při předchozí práci.

Vyhledání díla
Uživateli je dána možnost interaktivně vyhledávat soubory bez závislosti na automatickém vyhledávači. Uživatel zadá název souboru a případně zvolí uložení tohoto názvu souboru a výsledku vyhledávání do databáze. Po odeslání požadavku dojde k zobrazení nalezených děl.

Odeslání emailu k pirátskému dílu
Tato funkčnost zatím nebude implementována. Bude implementována až po úspěšném ověření funkčnosti aplikace. Uživateli bude dána možnost odeslat email informující server o výskytu pirátské kopie se žádostí o jeho odstranění. Emailová adresa bude součástí definice serveru.

Automatické hledání děl
Na serveru bude založen automatický proces, který bude vyhledávat soubory. Proces při automatickém spuštění zjistí z databáze uživatelů, zda nastal čas pro vyhledání jimi zadaných souborů. Pokud tento čas nastal, načte seznam jejich požadavků a spustí vyhledávání. Výsledek vyhledávání uloží do databáze. Po dokončení dojde k vyvolání Odeslání vyhledaných děl uživateli.

Odeslání vyhledaných děl uživateli
V případě, že automatický proces nalezl pro uživatele záznamy, které odpovídají jeho požadavkům, dojde k odeslání informačního emailu na adresu zadanou uživatelem. Informační email obsahuje počet nově nalezených souborů.

Zrušení účtu uživatele (zneplatnění účtu)
Administrátor aplikace má možnost zablokovat účet uživateli, který aplikaci zneužívá. Odebrání uživatelského přístupu je realizováno změnou hesla k účtu. Další možností je odmazání celého záznamu z databáze uživatelů.

Schválení návrhu uživatele na doplnění serveru pro hledání díla
Pokud správce aplikace obdrží od uživatele návrh na přidání nového vyhledávacího serveru, provede analýzu požadavku a rozhodne o přidání tohoto serveru. Je potřeba analyzovat strukturu HTML stránek serveru a rozhodnout o možnosti načtení výsledků hledání.

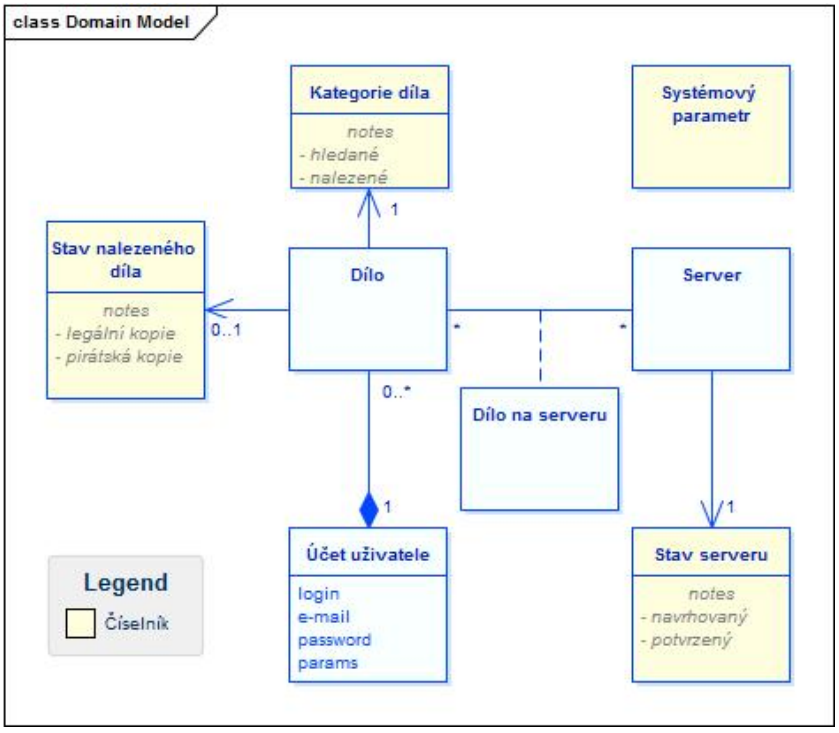
Nastavení parametrů pro automatické vyhledávání
Správce definuje časové intervaly vyhledávání, ze kterých si uživatelé vybírají frekvenci automatického spouštění vyhledávání.

Správa serverů pro automatické vyhledávání
Správce definuje seznam vyhledávacích serverů a vytváří algoritmus pro čtení a zápis výsledků hledání.

Kapitola 6

Doménový model

Doménový model nacházející se na obrázku zachycuje hlavní entity vyskytující se v aplikaci.



Obrázek 6.1: Doménový model

Účet uživatele
Entita reprezentující uživatele aplikace. Jejími atributy jsou přihlašovací jméno, email, přihlašovací heslo a vlastnosti vyhledávání pro konkrétního uživatele.

Dílo
Entita reprezentující soubor zadaný pro vyhledávání. Tato entita obsahuje dvě kategorie souboru. Hledaná díla jsou díla, která jsou zadaná do vyhledávače a nalezená díla, která již vyhledávač nalezl a uložil je do databáze. Takto nalezené soubory mohou nabývat dvou forem, legální kopie a nelegální kopie

Server
Entita reprezentující server určený na ukládání a vyhledávání souborů. Tento server může v aplikaci dosahovat dvou stavů-navrhovaný a potvrzený server. Navrhovaný server je takový, na který odeslal uživatel správci aplikace požadavek na přidání do vyhledávání. Pro potvrzený server je již implementován vyhledávací algoritmus.

Kapitola 7

Použité technologie

7.1 Java

Java je objektově orientovaný programovací jazyk, který vyvinula firma Sun Microsystems [2]. V současné době je vyvíjen společností Oracle. Je meziplatformně přenositelný na úrovni zdrojového i přeloženého kódu. Přeložený javový program běží v tzv. Java Virtual Machine (JVM).
Java EE (Java Platform, Enterprise Edition) je součást platformy Java určené pro vývoj a provoz podnikových aplikací a informačních systémů. Aplikace pro platformu Java EE jsou vyvíjeny na základě API a dalších fragmentů definovaných v jednotlivých specifikacích. Běhovým prostředím pro tyto aplikace je poté tzv. Aplikační Server [3].

7.2 Aplikační server

Aplikační server (Application server, někdy se používá zkratka APS) je pojem pro server - software, který je specializovaný pro provozování nějaké sdílené aplikace. Jedná se o softwarovou platformu, která sedí nad operačním systémem a zajišťuje základní služby pro provoz samotných aplikací.

Aplikační server je typickou součástí tzv. třívrstvé architektury, v rámci které se stará o samotný provoz aplikací, zejména o jejich tzv. business logiku.

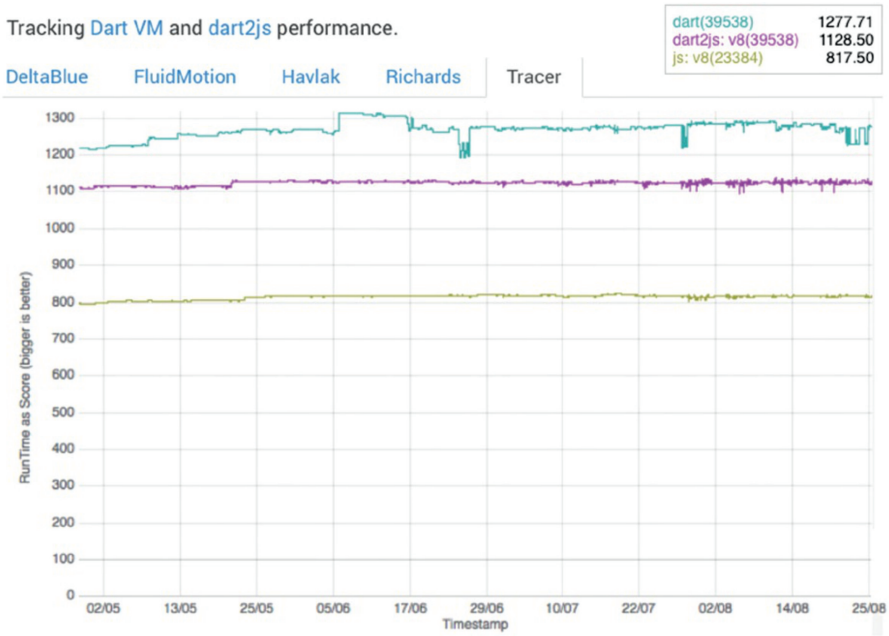
7.3 HTML

HTML(HyperText Markup language) je značkovací jazyk využívaný pro tvorbu webových stránek. Od jiných značkovacích jazyků se odlišuje tím, že obsahuje definovanou množinu značek(tzv. tagů).

7.4 Google Dart

Dart je open-source, strukturovaný a flexibilní programovací jazyk vyvíjený společností Google, který se orientuje zejména na vývoj webových aplikací [1].

Dart může být spuštěný v Dart virtual machine, která je součástí internetového prohlížeče Dartium, nebo může být pomocí kompilátoru dart2js zkompilován do javascriptu, který je podporován všemi běžnými internetovými prohlížeči. Výhodou Dartu je rychlost oproti javascriptu, která se projevuje, když je program spuštěn v Dart virtual machine, i když je zkompilován do javascriptu. Toto srovnání je vidět na přiloženém grafu 7.1.



Obrázek 7.1: Srovnání Google Dart a javascriptu, převzato z Web Programming with Dart [1]

7.5 Twitter Bootstrap

Twitter Bootstrap je velmi jednoduchý a volně dostupný soubor nástrojů pro vytváření moderního webu a responzivních webových aplikací [4]. Nabízí podporu nejrůznějších webových technologií HTML, CSS, JavaScript a mnoho prvků, které je možné snadno implementovat do své stránky [5]. Současně s Bootrstap je používána implementace pro Google Dart s názvem Bootjack.

7.6 CSS

Kaskádové styly je jazyk sloužící pro grafickou úpravu webových stránek. Definuje vzhled jednotlivých značek HTML dokumentu.

7.7 Intel XDK

Intel XDK je vývojové prostředí, které slouží pro tvorbu multiplatformních mobilních aplikací [6]. Tyto aplikace jsou tvořeny pomocí HTML5 a převedeny do mobilní aplikace.

7.8 SQL

SQL(Structured Query Language) je standardizovaný strukturovaný dotazovací jazyk, který je používán pro práci s daty v relačních databázích [7].

7.9 JDBC

Java Database Connectivity je API v programovacím jazyce Java, které definuje rozhraní pro přístup k relačním databázím [8]. Pro přístup ke konkrétnímu databázovému serveru je potřeba JDBC ovladač, který poskytuje tvůrce databázového serveru [9].

Kapitola 8

Řešení

Jedním ze základních požadavků na aplikaci byla přenositelnost na různé platformy a libovolné velikosti zobrazovacího zařízení. Zároveň bylo požadováno umožnit spouštění vyhledávání v pravidelných časových intervalech. Proto jsem zvolil řešení pomocí responzivní HTML webové aplikace. To dovoluje spouštět aplikaci na jakémkoli zařízení s internetovým prohlížečem. Vzhledem k tomu jsem se rozhodl samotné vyhledávání implementovat jako aplikaci umístěnou na severu. Díky tomu má klientská aplikace minimální nároky na hardware a výrazně se snížily i nároky na internetové připojení, protože klientská aplikace pouze stahuje výsledky vyhledáváni ze serveru, místo aby sama prohledávala stránky, zda obsahují pirátské soubory.

Aplikace je založena na architektuře klient-server. Toto řešení umožňuje přenést zátěž z klientské aplikace na server a omezit datové přenosy na mobilním zařízení. Výhodou je také implementace pouze jednoho vyhledávacího algoritmu na serveru, ke kterému mohou přistupovat klienti pracující na různých platformách. Klient má v tomto řešení pouze zobrazovací funkci.

8.1 Server

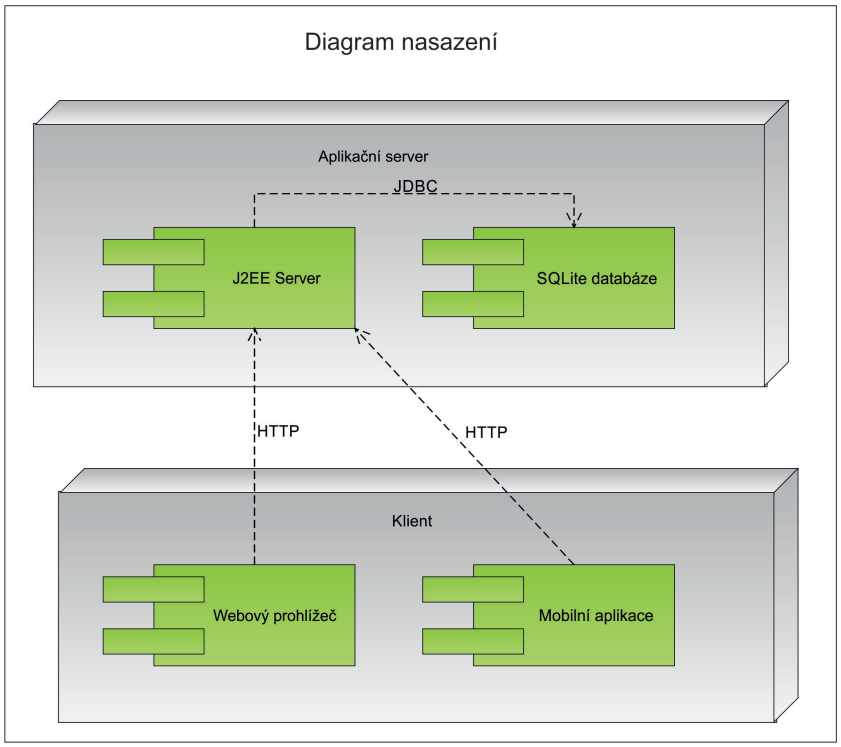
Representational State Transfer – REST je styl softwarové architektury, určený pro distribuované hypermediální systémy jako je World Wide Web [10].

Pro realizaci serverové části aplikace pro vyhledávání jsem zvolil webovou aplikaci na platformě jazyka Java EE. Toto řešení umožňuje nasazení aplikace na libovolný operační systém, pro který existuje implementace **JRE**(Java Runtime Environment). To splňují všechny serverové operační systémy založené na Linuxu a Windows. Základem serverové aplikace jsou servlety, což jsou třídy obsluhující požadavky http protokolu. Servlety nejsou spustitelné aplikace, ale žijí uvnitř prostředí, zvaného servlet container. Je definováno

rozhraní mezi servlet containerem a webovou aplikací tvořenou servlety, tzv. Servlet API, které má různé verze - v současnosti se používají verze 2.5 a 3.0 [11]. Existují různé implementace servlet containerů, např. Apache Tomcat, Jetty. Ve své práci jsem se rozhodl pro použití servlet containeru GlassFish Server.

8.2 Klient

Klientská část aplikace komunikuje se serverem. Lze měnit nastavení serveru, definovat vyhledávané soubory a zobrazovat výsledky hledání na serveru.
Klientská aplikace je realizována pomocí HTML, Google Dart a Javascriptu. Je implementována jako responzivní, aby bylo možné obrazovky aplikace zobrazit na smartphonu, tabletu i stolním počítači. Rosponzivní design je způsob stylování HTML dokumentu, který zaručí, že zobrazení stránky bude optimalizováno pro všechny druhy nejrůznějších zařízení (mobily, notebooky, netbooky, tablety, ard.) [12].
Responzivní design je řešen pomocí frameworku Twitter Bootstrap v kombinací s jeho implementací pro programovací jazyk Dart, která má název Bootjack.
Cílem frameworku je převzetí typických problémů dané oblasti, čímž se usnadní vývoj tak, aby se návrháři a vývojáři mohli soustředit pouze na své zadání.[13]



Obrázek 8.1: Diagram nasazení

Kapitola 9

Realizace

9.1 Popis klientské aplikace

9.1.1 Tvorba klientské aplikace

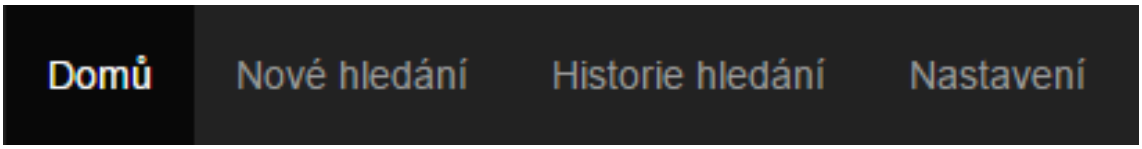
Prezentační vrstva je vytvořena jako webová aplikace. Pro její vytvoření byly použity programovací jazyky HTML, CSS a Google Dart. Výhodou tohoto řešení je skutečnost, že se vytvoří pouze jedna aplikace, která je spustitelná na všech běžných operačních systémech a není nutné vytvářet více speciálních aplikací pro každý operační systém.
Aplikace je optimalizována pro použití na zobrazovacích zařízeních s různým rozlišením a různou velikostí obrazovky. Z tohoto důvodu je vytvořena jako responzivní-automaticky se přizpůsobující zobrazovacímu zařízení.
Pro tvorbu responzivního uživatelského rozhraní je použit nástroj Twitter Bootstarp a jeho implementace pro programovací jazyk Dart s názvem Bootjack.
Pomocí jazyka Google Dart je vytvořena hlavní logika klientské aplikace. Google Dart umožňuje číst hodnoty zadané uživatelem do editačních prvků na HTML stránce. Ze zadaných hodnot sestaví příkaz pro serverovou část aplikace. Komunikace se serverovou částí aplikace je zajištěna pomocí http protokolu.

```
var url = "http://" + hostName
      + ":" + port
      + "/PirateSearch-war/CommunicationServlet?delete="
      + id;
var request = HttpRequest.getString(url);
```

Po odeslání příkazu a obdržení odpovědi ze serveru se postará o její zpracování a zobrazení přijatých informací na výsledné HTML stránce.

9.1.2 Uživatelské rozhraní

Uživatelské rozhraní je tvořeno čtyřmi základními obrazovkami, do kterých se vstupuje pomocí nabídky umístěné v horní části obrazovky.



Obrázek 9.1: Menu klientské aplikace

9.1.3 Domů - úvodní obrazovka

Úvodní obrazovka, přístupná také přes nabídku "Domů", obsahuje základní možnosti programu a slouží k registraci nových uživatelů do systému a k přihlášení již existujících uživatelů. Přihlášeným uživatelům také dává možnost ukončit práci s programem odhlášením.

9.1.3.1 Registrace

Při vytváření nového přístupového účtu musí uživatel zadat následující udaje:

- uživatelské jméno - musí být unikátní v rámci aplikace
- heslo - nesmí být prázdné a musí se zadat dvakrát pro ověření správnosti
- emailová adresa - na tuto adresu program odesílá informace o nalezených souborech

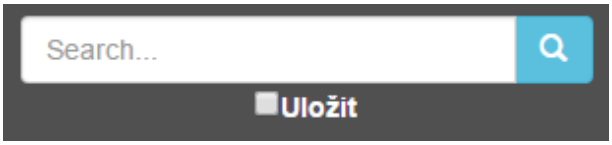
Po úspěšné registraci může uživatel začít využívat funkčnost aplikace.

9.1.3.2 Přihlášení

Pokud je uživatel v systému již registrován, může se přihlásit zadáním uživatelského jména a hesla. Po úspěšném přihlášení může pracovat s programem. Pokud se přihlášení nezdařilo, je zobrazena chybová hláška.

9.1.4 Nové hledání

Na obrazovce Nové hledání má uživatel možnost spustit vyhledávání souboru zadaného do textového pole. Pod textovým polem se nachází zaškrtávací políčko, které určuje, zda má být zadaný název souboru a výsledek vyhledávání uloženy do databáze hledaných a nalezených souborů.



Obrázek 9.2: Textové pole pro zadání názvu souboru

Po odeslání příkazu je zobrazena tabulka s nalezenými soubory. Hledání souborů probíhá na všech serverech, pro které je v aplikaci definován vyhledávací algoritmus. Vzhled tabulky se automaticky přizpůsobuje velikosti a orientaci zobrazovacího zařízení, aby byla zachována přehlednost.

9.1.5 Historie hledání

Obrazovka Historie hledání slouží uživateli k zobrazení souborů, které byly nalezeny automaticky spuštěným vyhledáváním na serverové části aplikace, i souborů nalezených při předchozích manuálně spuštěných vyhledávání, u kterých uživatel zadal uložení výsledku do historie.
Uživatel má možnost před odesláním příkazu pro zobrazení historie vyhledávání omezit výsledný seznam nalezených souborů zadáním filtru na vyhledávací servery, uložené názvy souborů, typy souborů a příznaku, zda již uživatel soubory zobrazil při předchozí práci.

Obrázek 9.3: Nastavení zobrazení výsledku vyhledávání

Po odeslání příkazu je zobrazena tabulka, která obsahuje nalezené soubory podle zvolených kritérií.
Během výběru souborů pro zobrazení má uživatel možnost zadat nové názvy souborů pro vyhledávání a také smazat již zadané názvy souborů.

9.1.6 Nastavení

V obrazovce Nastavení má uživatel možnost změnit nastavení svého účtu. Uživatel může změnit emailovou adresu, na kterou jsou mu zasílány výsledky po automaticky spuštěném vyhledávání. Email je odesílán pouze v případě, když uživatel v nastavení vybere možnost pro automatické odesílání emailu.

Další součástí nastavení je možnost vybrat frekvenci spouštění automatického vyhledávání z předem nadefinovaných hodnot. Pod touto možností se nachází možnost změnit přihlašovací heslo. Poslední částí nastavení je možnost odeslat administrátorovi systému požadavek na přidání nového serveru do vyhledávání. Tento požadavek bude administrátorovi odeslán emailem.

9.2 Popis serverové aplikace

Serverová část aplikace je vytvořena v Java Enterprise Edition. Obsahuje dva moduly war a ejb.
Ejb modul obsahuje logiku aplikace a zajišťuje pravidelné vyhledávání souborů. V tomto modulu je realizováno načtení definice serverů, na kterých aplikace umí vyhledávat, a spuštění vlastního vyhledávání, a odesílání emailů s informacemi o nalezených souborech. Důležitou částí aplikace je automatické spouštění vyhledávání souborů v intervalu, který si uživatel zvolil. Aplikace každou hodinu zjistí z databáze, pro které uživatele má provést hledání. Tato cyklická činnost je zajištěna plánovačem úloh (scheduler), který je v Jave vytvořen pomocí třídy ScheduleExpression. Po vytvoření instance této třídy je nutné nastavit, v jakých intervalech má spouštět určitou činnost.

Pro opakování každou hodinu vypadá nastavení takto:

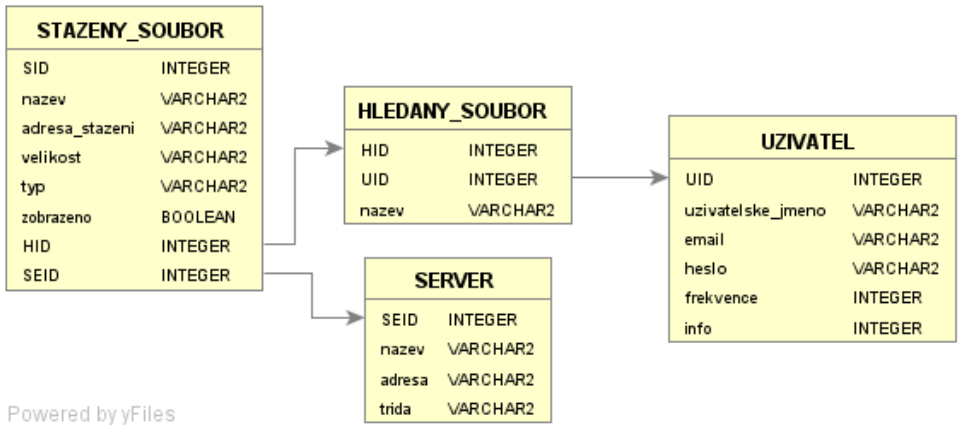
```
ScheduleExpression schedExp = new ScheduleExpression();
schedExp.start(new Date());
schedExp.second("0");
schedExp.minute("0");
schedExp.hour("*");
schedExp.dayOfMonth("*");
schedExp.month("*");
schedExp.year("*");
```

War modul obsahuje dva servlety, které zajišťují přístup k serverovým funkcím. Servlet StartServer slouží správci aplikace k načasování automatického spouštění vyhledávání. Servlet CommunicationServlet je hlavním přístupovým bodem klientské aplikace k serverovým funkčnostem. Klient komunikuje se servletem pomocí http protokolu.

9.3 Databáze

Pro realizaci databázové části jsem si vybral relační databázi SQLite. Ta na rozdíl od databází založených na principu klient-server, kde je databázový server spuštěn jako samostatný proces, je pouze malá knihovna, která se přilinkuje k aplikaci a pomocí jednoduchého rozhraní ji lze začít využívat [14]. Samotná databáze je uložena v souboru *.db [15].

Databáze obsahuje celkem 4 tabulky. HLEDANY_SOUBOR, SERVER, STAZENY_SOUBOR a UZIVATEL. Tabulka UZIVATEL reprezentuje uživatelský účet. U každého uživatele je uloženo jeho uživatelské jméno, hash hesla, email, na který se odesílají záznamy o nalezených souborech a frekvence, kterou si uživatel nastavil pro vyhledávání souborů. Každý uživatel má uložené záznamy v tabulce HLEDANY_SOUBOR. Reprezentuje soubory, které uživatel zadal pro vyhledávání. Tato tabulka obsahuje identifikátor hledaného souboru a jeho název. Tabulka SERVER zastupuje jednotlivé internetové stránky, které potenciálně obsahují pirátské kopie souborů. Obsahuje identifikační číslo serveru, název, jeho internetovou adresu a název třídy, ve které je implementován algoritmus pro stahování souborů. STAZENY_SOUBOR reprezentuje konkrétní výsledky vyhledávání. Obsahuje atributy identifikátor souboru, jeho název, internetovou adresu, na které je soubor umístěn, velikost na disku, identifikátor serveru, na kterém se soubor nachází, a informaci, zda už byl zobrazen uživatelem.



Obrázek 9.4: Model databáze

Kapitola 10

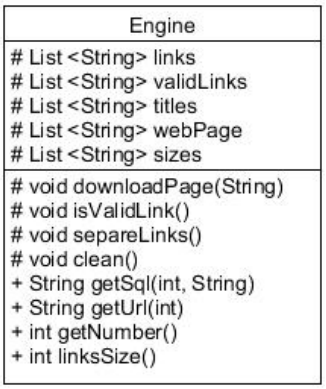
Stahování výsledků ze serverů

Každý server má vlastní logiku vytváření stránek s výsledky vyhledávání. Proto je potřeba pro každý server implementovat vlastní algoritmus, který dokáže rozklíčovat a vyhodnotit stažený HTML kód.

Aby bylo možné vyhledávací proces na všech serverech co nejvíce sjednotit, byla vytvořena třída Engine, která je předkem pro všechny třídy, které reprezentují jednotlivé servery.

Třída Engine definuje základní rozhraní pro vyhledávání na serverech a obsahuje implementace základních metod, které jsou pro většinu serverů společné, a metody, které slouží k vracení výsledků v jednotném tvaru.

Každý server je v programu reprezentován třídou, která je potomkem třídy Engine. V každé této třídě je potřeba doimplementovat individuální algoritmus, který je dán způsobem odesílání požadavků na server a strukturou HTML kódu vrácené stránky. Je potřeba staženou HTML stránku analyzovat a získat požadované údaje o nalezených souborech.



Obrázek 10.1: Třída Engine

10.1 Popis principu odesílání požadavků na sever

U každého serveru je třeba zjistit a přesně určit, jakým způsobem je třeba zasílat dotaz na vyhledání souboru. Struktura dotazu je vždy stejná. Jedná se o http příkaz, který obsahuje internetovou adresu serveru, klíčové slovo pro hledání a název hledaného souboru. Název hledaného souboru se musí před přidáním do dotazu převést do správného formátu, který se liší pro každý server. Například se jedná o rozdílnou interpretaci mezery a českých znaků.

Několik příkladů vyhledávacích příkazů pro nalezení výrazu "a bc čd" pro různé servery:

```
http://www.edisk.cz/vyhledat/a_bc_cdd
http://uloz.to/hledej?q=a+bc+čdď
http://datoid.cz/s/a-bc-cdd
http://megarapid.cz/vyhledavani/?q=a+bc+%C4%8Dd%C4%8F
```

10.2 Algoritmus pro vyhodnocování výsledků

Po odeslání dotazu na server je možné stáhnout a analyzovat výsledek vyhledávání.

downloadPage(String url)
Tato metoda slouží pro stažení HTML kódu stránky s výsledky vyhledávání. Metoda čte zdrojový kód stránky po řádcích, které ukládá do seznamu webPage. Z takto uložené stránky lze později získat názvy vyhledaných souborů, jejich internetové adresy a další informace o sdílením souboru. Tato metoda neslouží k vyhodnocování HTML kódu.

separeLinks()
Uložená stránka obsahuje více internetových adres než jen odkazy na soubory. Všechny tyto adresy se od kódu celé stránky oddělí v metodě separeLinks podle klíčového slova *href*, které je v HTML značkou pro odkaz. Tyto odkazy jsou uloženy v seznamu links.

Metody ***downloadPage()*** a ***separeLinks()*** jsou implementovány na předku Engine a jsou použity pro všechny dosud zavedené servery. Pouze pro server uloz.to musely být tyto metody přepsány na úrovni potomka.

isValidLink()
Ze všech adres je nutné oddělit adresy, které obsahují odkaz na soubor. To obstarává metoda s názvem isValidLink. Tento algoritmus je rozdílný pro každý server, a proto je metoda na každém potomku přepsána. Pro implementaci

této metody je nutné zjistit, jakým způsobem je v HTML kódu reprezentován každý soubor. Například na serveru megarapid.cz je v tabulce výsledků každému souboru přiřazen identifikátor "data-file-id". Pokud odkaz obsahuje klíčový řetězec reprezentující soubor, je uložen do seznamu validLinks.

clean()
Metoda clean() je volána z metody isValidLink() a slouží k vytěžení informací odkazu. Jedná se o jméno souboru, internetovou adresu, kde je soubor uložen, a velikost.

Alogoritmus pro stažení HTML stránek, jejich čtení a získávání informací je pro většinu serverů podobný. Ze všech serverů zpracovaných v této práci je odlišný pouze server Uloz.to. Vzhledem k tomu, že tento server je v České republice jeden z nejpoužívanějších, byla mu věnována největší pozornost a byl navržen individuální postup, jak získávat potřebné informace.

10.3 Vyhodnocování ze serveru Uloz.to

Struktura zdrojového kódu serveru Uloz.to se velice liší od ostatních serverů. Je úmyslně navržena tak, aby co nejvíce ztížila vyhledávání internetovým robotům. Proto vyhledávání na uloz.to není podporováno na specializovaných stránkách jako jsou esoubory.cz nebo superload.cz. Vyhledávání na Uloz.to není na těchto stránkách buď vůbec nabízeno, nebo výsledky hledání nezobrazují.
Stažený HTML kód stránky s výsledkem hledání neobsahuje čitelné informace o souborech, pouze javascript kód, který v prohlížeči vygeneruje seznam souborů. Tento postup je na Uloz.to zvolen z důvodu zabránění použití vyhledávacích robotů.
O zobrazování výsledků se stará tento javascript kód:

```
$(document).ready(function() {
        var loc = document.location.toString();
        $.ajax({
            type: 'GET',
            url: loc.replace(/(?:([\?&])q=)([^\&]*)/,
                 '$1q=$2%20'),
            dataType: 'html',
            success: function(data) {
                $('#search-content').html(data);
                if (typeof ad != "undefined")
                {
                    ad.walk('#search-content
                        .chessFiles li', 'data-icon');
                    ad.walk('#search-content
```

```
                    .textFiles tr', 'data-icon');
            }
            thumbRotationRegister();
            paginateJs();
}}}
```

Aby se javascript kód spustil, musí se odeslat požadavek na server s potřebnými vlastnostmi. To zajistí, že ve stažené HTML stránce je javascript kód nahrazen šifrovanými informacemi, ze kterých lze postupným dešifrováním získat HTML kód se seznamem souborů.

Vlastnosti http požadavku:

```
connection.setRequestProperty("X-Requested-With",
                              "XMLHttpRequest");
connection.setRequestProperty("Referer", url);
connection.setRequestProperty("Cookie",
                              "uloz-to-id=1561277170;");
```

Jednotlivé nalezené soubory jsou definovány elementem li a atributem data-icon, který obsahuje identifikátor pro další použití.

```
<li data-icon="95b74af61b53ee30f8a67480dbb8ce82"></li>
```

Stránka dále obsahuje javascript proměnou var kn, ve které jsou uložené zašifrované informace o souborech.

```
var kn = {"95b74af61b53ee30f8a67480dbb8ce82":
"c43ec9a07adedb0baac01204fd891f110bcc113055d7d3dd", ...}
```

Z proměnné kn se podle identifikátoru souboru definované pomocí data-icon získá zašifrovaný blok, který obsahuje HTML kód s konkrétními informacemi o souboru. Z těchto informací lze získat název souboru, velikost a zašifrovaný odkaz. Odkaz se získá opětovným dešifrováním.

Pro dešifrování obsahu stránek je použita java knihovna, která umožňuje spouštět javascriptový kód. Jedná se o knihovnu ScriptEngine. ScriptEngine musí být před použitím inicializován hodnotami, které jsou obsaženy v souboru jquery.min.js, na nějž odkazují HTML stránky uloz.to.

```
engine.eval("var decrypted =
 ad.decrypt(kn[\"" + keys.get(i) + "\"]);");
```

Kapitola 11

Přidávání nových serverů

Důležitou vlastností aplikace je možnost snadného rozšíření vyhledávání o další servery podle přání uživatelů. Snadné rozšíření je dáno využitím jednotného aplikačního rozhraní pro všechny servery. Toto rozhraní je definováno ve třídě Engine, která je předkem pro všechny třídy reprezentující vyhledávací servery.
Pro vytvoření instance serveru při vyhledávání je využito reflexe (Java Reflection). Reflexe je schopnost programovacího jazyka zjistit za běhu informace o určitém objektu [16]. Reflexe umožňuje zjistit typ objektu až za běhu aplikace.

11.1 Postup přidání serveru

Pro přidání nového serveru je potřeba vytvořit novou třídu, jejímž předkem je třída Engine a naimplementovat vyhledávací algoritmus. Následně je potřeba tuto třídu zkompilovat do knihovny (jar) a tuto knihovnu přidat do classpath aplikace. Název třídy pro vyhledávací server je uložen v databázi. Aplikace vytváří instance tříd vyhledávacích serverů načtením těchto názvů.

Příklad použití:

```
Class c = Class.forName("cz.cvut.pirateSearch.Ulozto");
Engine e = (Engine) c.newInstance();
```

Kapitola 12

Odesílání emailů

Aby uživatel nemusel sám kontrolovat výsledek automaticky spouštěného vyhledávání v klientské aplikaci, je informován o výsledku vyhledávání odesláním emailu na definovanou adresu. Tato zpráva obsahuje počet nově nalezených souborů.
Aplikace také umožňuje odeslat email se žádostí o přidání nového serveru do vyhledávání správci aplikace.
Pro odeslání emailu se využívá Java knihovna javax.mail. Tato knihovna obsahuje všechny prvky potřebné pro práci s emailem.
V aplikaci je využívána emailová schránka definovaná na serveru yahoo.com.

Konfigurace parametrů v aplikaci pro připojení k serveru yahoo.com:

```
Properties properties = System.getProperties();
properties.put("mail.smtp.starttls.enable", "true");
properties.put("mail.smtp.host", "smtp.mail.yahoo.com");
properties.put("mail.smtp.user", from);
properties.put("mail.smtp.password", pass);
properties.put("mail.smtp.port", "587");
properties.put("mail.smtp.auth", "true");
```

Použití:

```
SendEmail se = new SendEmail();
String subject = "Požadavek na přidání serveru";
String message = "Uživatel " + userID
      + " poslal požadavek o přidání serveru "
      + request.getParameter("addServer") + ".";
se.send("pirateSearch.user@gmail.com", subject, message);
```

Kapitola 13

Sestavení

13.1 Sestavení klientské aplikace

Logická část klientské aplikace je vytvořena v programovacím jazyku Google Dart. Aby mohl být vytvořený kód spuštěn v internetovém prohlížeči je nutná podpora Dart virtual machine. Tuto podporu ale obsahuje pouze prohlížeč Dartium. Proto je nutné aplikaci převést do mnohem běžnějšího jazyka Javascript, který je podporován ve všech internetových prohlížečích.
Sestavování probíhá z příkazového řádku pomocí programu **pub**. Nejprve je nutné stáhnout všechny balíčky (packages), na kterých je aplikace závislá a které jsou uvedeny v konfiguračním souboru *pubspec.yaml*. To se provede příkazem

```
pub install
```

Pro samotné sestavení slouží příkaz

```
pub build
```

Po jeho provedení se ve složce projektu vytvoří nový adresář s názvem **build**, který obsahuje kompletní webovou aplikaci. Všechny soubory jazyka Dart jsou převedeny do jazyka Javascript pomocí kompilátoru s názvem dart2js. Pro každý soubor **.dart** vznikne **.dart.js**.
Takto sestavenou webovou aplikaci je možné spustit v jakémkoli internetovém prohlížeči.

13.2 Sestavení serverové aplikace

Serverová část aplikace byla vytvořena ve vývojovém prostředí NetBeans. Toto prostředí obsahuje funkčnost pro vytvoření instalačního balíčku. Pomocí funkce **build** je vytvořen soubor *PirateSearch.ear*. Tento soubor obsahuje všechny potřebné knihovny a sestavené moduly ejb a war.

Kapitola 14

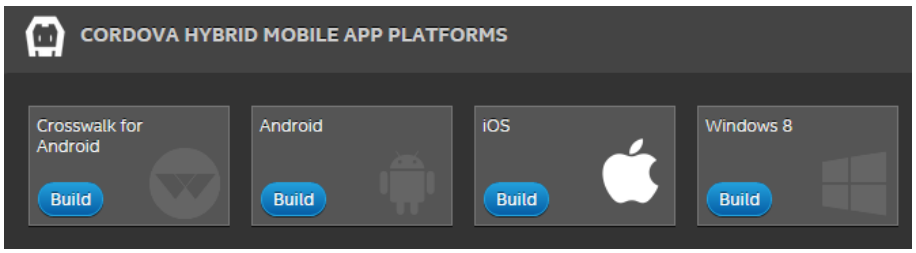
Tvorba mobilní aplikace

14.1 Popis

Pro tvorbu mobilních aplikací spustitelných na smartphonech a tabletech s operačními systémy Android a iOS je použito vývojové prostředí Intel XDK. Tyto aplikace jsou vytvořeny v HTML, CSS a javascriptu a převedeny pomocí Intel XDK do mobilní aplikace pro konkrétní platformu.
Produktem Intel XDK je instalační balíček apk pro Android a ipa pro iOS. Tento balíček lze umístit do obchodů s mobilními aplikacemi Google Play a Apple Store.
Tyto aplikace lze běžným způsobem nainstalovat na mobilní zařízení a aplikace se pak chovají jako běžné aplikace napsané v nativním jazyce mobilní platformy. Pomocí platformy Apache Cordova je možné zpřístupnit funkce mobilních zařízení-fotoaparát, senzory, telefonní seznam převedeným webovým aplikacím [17].
Mobilní aplikace vytvořené tímto postupem se nazývají hybridní aplikace. Nemají takový výpočetní výkon jako nativní aplikace. Nižší výkon není pro zde navrhovanou aplikaci omezením, neboť má jen zobrazovací funkci.

14.2 Postup

Po vytvoření HTML aplikace lze zdrojový kód aplikace naimportovat do projektu ve vývojovém prostředí Intel XDK. Takto naimportovanou aplikaci lze zobrazit v emulátoru pro různá mobilní zařízení.
Po kontrole funkčnosti lze zadat požadavek na sestavení aplikace pro různé platformy.



Obrázek 14.1: Nabídka Intel XDK

Kapitola 15

Testování

Důležitou součástí vývoje software je jeho důkladné testování. To slouží k odhalení chyb a ověření správné funkčnosti aplikace.
Existuje několik základních typů testování:

- Assembly tests (Testování programátorem)
- Unit tests (Testování jednotek)
- FAT (Funkční testy)
- Integrační testování
- Systémové testování
- UAT (Akceptační testování)

Při testování serverové části aplikace jsem se zaměřil na systémové testy - testování celé aplikace jako jednoho funkčního celku. Primárně je testováno HTTP spojení a ověření správnosti přijatých výsledků. Tímto způsobem je otestována většina funkcionalit zahrnutých v serverové aplikaci. Testy byly provedeny pomocí knihovny JUnit.
Je simulován klient, který pomocí http komunikuje se serverem.
Je testováno úspěšné vytvoření nového uživatele, neúspěšné založení uživatele snažícího se zaregistrovat do aplikace pomocí již existujícího jména, úspěšné přihlášení uživatele, neúspěšné přihlášení uživatele pod neexistujícím jménem nebo chybným heslem, zobrazení předem založených záznamů v tabulce výsledků, nalezení testovacího souboru *pirateCopyTest.txt* na vyhledávacím serveru Edisk.cz.

Příklad metody, která testuje přihlášení neexistujícího uživatele:

```
@Test
public void testLogin() throws IOException{
```

```
    downloadHTTP("http://10.0.0.140:8080/PirateSearch-war
                 /CommunicationServlet?logOut=ok");
    String result = downloadHTTP("http://10.0.0.140:8080
                 /PirateSearch-war/CommunicationServlet
                 ?login=uzivatel123&pass=heslo123");
    assertEquals("user", result);//neexistujici uzivatel
}
```

Všechny tyto testy proběhly v pořádku.

Kapitola 16

Nasazení

Aby mohla být serverová část aplikace nainstalována, musí pro operační systém existovat implementace JDK 8 a implementace aplikačního serveru GlassFish. Tu je možné volně stáhnout na adrese `https://glassfish.java.net/download-archive.html` pro operační systémy windows a unix.
Instalační postup:

1. Instalace aplikačního serveru Glassfish
2. Vytvoření nové domény serveru příkazem:

   ```
   asadmin create-domain --adminport 4848 psDomain
   ```

3. Spuštění aplikačního serveru příkazem:

   ```
   asadmin start-domain psDomain
   ```

4. Spuštění administrátorské konzole v internetovém prohlížeči zadáním adresy `http://hostname:4848`
5. Instalace aplikace na aplikační server z balíčku *PirateSearch.ear*

Takto nainstalovanou aplikaci lze spustit v internetovém prohlížeči na adrese `http://hostname:post/PirateSearch-war/index.html`

Závěr

Cílem mé práce bylo zjistit požadavky nakladatele na aplikaci, na jejich základě navrhnout a vytvořit softwarový systém, který napomůže při vyhledávání pirátských kopií děl na internetu.
Výsledkem je serverová aplikace napsaná v programovacím jazyce Java, která automaticky prochází vybrané servery určené ke sdílení souborů a vyhledává na nich definované soubory. S touto aplikací komunikuje webová aplikace napsaná pomocí jazyků HTML a Google Dart.

Ve své práci jsem se zaměřil na splnění funkčních a nefunkčních požadavků. Požadavek F1 byl vyřešen pomocí plánovače úloh (scheduler), který každou hodinu zjistí, pro které uživatelské účty má spustit hledání. Pro splnění F2 byla použita java knihovna javax.mail, pomocí které jsou odesílány emaily uživatelům aplikace z emailové schránky serveru yahoo.com. F3 Možnost registrace nového uživatele je poskytnuta na domovské stránce klientské aplikace. Každý registrovaný uživatel má možnost v klientské aplikaci na stránce Historie hledání zobrazit, jaké má aktuálně zadané soubory pro vyhledávání, a může je libovolně přidávat nebo mazat. Tím je splněn požadavek F4. Aby byl splněn funkční požadavek F5, obsahuje klientská aplikace možnost zobrazit výsledky vyhledávání v responzivní tabulce se všemi důležitými informacemi o souborech. Poslední funkční požadavek F6 je realizován v záložce Nastavení klientské aplikace, kde má uživatel možnost vybrat si z předvolených frekvencí pro automatické spouštění vyhledávání.
Jednoduché ovládání zadané v nefunkčním požadavku N1 je řešeno přehledným uživatelským rozhraním a je dbáno na to, aby u každého ovládacího prvku bylo jasné, k čemu slouží. Přizpůsobení uživatelského rozhraní různým velikostem zařízení z požadavku N2 zajišťuje responzivnost aplikace, která je realizována pomocí frameworku Twitter Bootstrap. Rozšiřitelnost a modifikovatelnost, která je zadaná v požadavku N3, se týká hlavně přidávání nových serverů do vyhledávání a úprav jejich vyhledávacího algoritmu. To je zajištěno reflexí v jazyku Java. N4 Přenositelnost na různé platformy je zajištěna tím,

že klientská část práce je realizována jako webová aplikace.

Náměty na další vývoj aplikace

- Zvýšení bezpečnosti komunikace klient-server přechodem na zabezpečený protokol https
- Odstranění nutnosti přihlašovat se při každém použití aplikace na stejném zařízení - využití identity uložené v mobilním zařízení
- Přidání možnosti u každého vyhledávaného souboru určit, jaký typ souboru se má vyhledávat
- Možnost manuálního odeslání žádosti o odstranění pirátské kopie ze serveru
- Identifikace odstraněných souborů z vyhledávacích stránek - jedná se o soubory, které byly nalezeny vyhledávačem v předchozích cyklech, ale následně byly odstraněny a vyhledávač je již nenalezl

Literatura

[1] Belchin, M.; Juberias, P.: *Web Programming with Dart*. Apress, první vydání, 2015, ISBN 978-1-484205-57-0.

[2] Eckel, B.: *Myslíme v jazyku Java*. Grada, první vydání, 2001, ISBN 80-247-9010-6.

[3] Goncalves, A.: *Beginning Java EE 7*. Apress, první vydání, 2013, ISBN 978-1-4302-4626-8.

[4] w3schools: *Bootstrap 3 [online]*. [cit. 2015-03-30]. Dostupné z: `http://www.w3schools.com/bootstrap/`

[5] Bootstrap: *Designed for everyone, everywhere [online]*. [cit. 2015-03-30]. Dostupné z: `http://getbootstrap.com/`

[6] intel Developer Zone: *Intel XDK [online]*. [cit. 2015-04-28]. Dostupné z: `https://software.intel.com/en-us/html5/tools`

[7] Krokodýlovy databáze: *SQL [online]*. [cit. 2015-04-9]. Dostupné z: `http://krokodata.vse.cz/SQL/SQL`

[8] JDBCtutorial.com: *Java JDBC Tutorial [online]*. [cit. 2015-04-3]. Dostupné z: `http://www.jdbc-tutorial.com/`

[9] ORACLE: *JDBC [online]*. [cit. 2015-04-3]. Dostupné z: `http://docs.oracle.com/javase/1.5.0/docs/guide/jdbc/`

[10] Softec: *REST – moderní alternativa programování webových aplikací [online]*. [cit. 2015-03-24]. Dostupné z: `http://www.softec.cz/aktuality/rest-moderni-alternativa-programovani-webovych-aplikaci.html`

[11] kore.fi.muni.cz: *VelkÚvod do webových aplikací [online]*. [cit. 2015-04-2]. Dostupné z: `https://kore.fi.muni.cz/wiki/index.php/%C3%9Avod_do_webov%C3%BDch_aplikac%C3%AD`

[12] LaGrone, B.: *HTML5 and CSS3 Responsive Web Design Cookbook*. Press, první vydání, 2013, ISBN 978-1-84969-544-2.

[13] ROOT.CZ: *Velký test PHP frameworků [online]*. [cit. 2015-04-20]. Dostupné z: `http://www.root.cz/clanky/velky-test-php-frameworku-2008/`

[14] ROOT.CZ: *SQLite - ultra lehké sql [online]*. [cit. 2015-04-12]. Dostupné z: `http://www.root.cz/clanky/sqlite-ultra-lehke-sql/`

[15] BANAN.CZ: *SQLite 1.díl, O co jde [online]*. [cit. 2015-04-12]. Dostupné z: `http://www.banan.cz/serialy/SQLite/SQLite-1-dil-O-co-jde/`

[16] Kiszka, B.: *1001 TIPŮ A TRIKŮ PRO PROGRAMOVÁNÍ V JAZYCE Java*. Press, první vydání, 2003, ISBN 80-7226-989-5.

[17] AndroidFórum: *Naprogramujte si android aplikaci bez znalosti Javy [online]*. [cit. 2015-04-27]. Dostupné z: `http://androidforum.cz/naprogramujte-si-android-aplikaci-bez-znalosti-javy-t64675.html`

Příloha **A**

Seznam použitých zkratek

HTML HyperText Markup Language

CSS Cascading Style Sheets

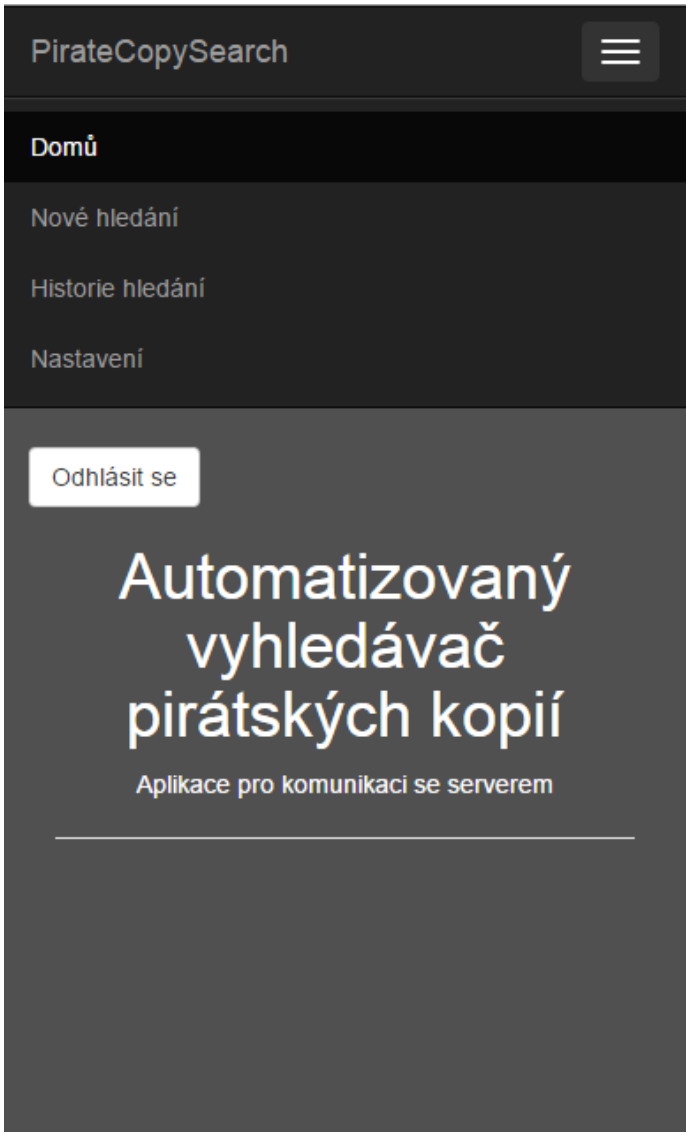
JDBC Java Database Connectivity

SQL Structured Query Language

REST Representational State Transfer

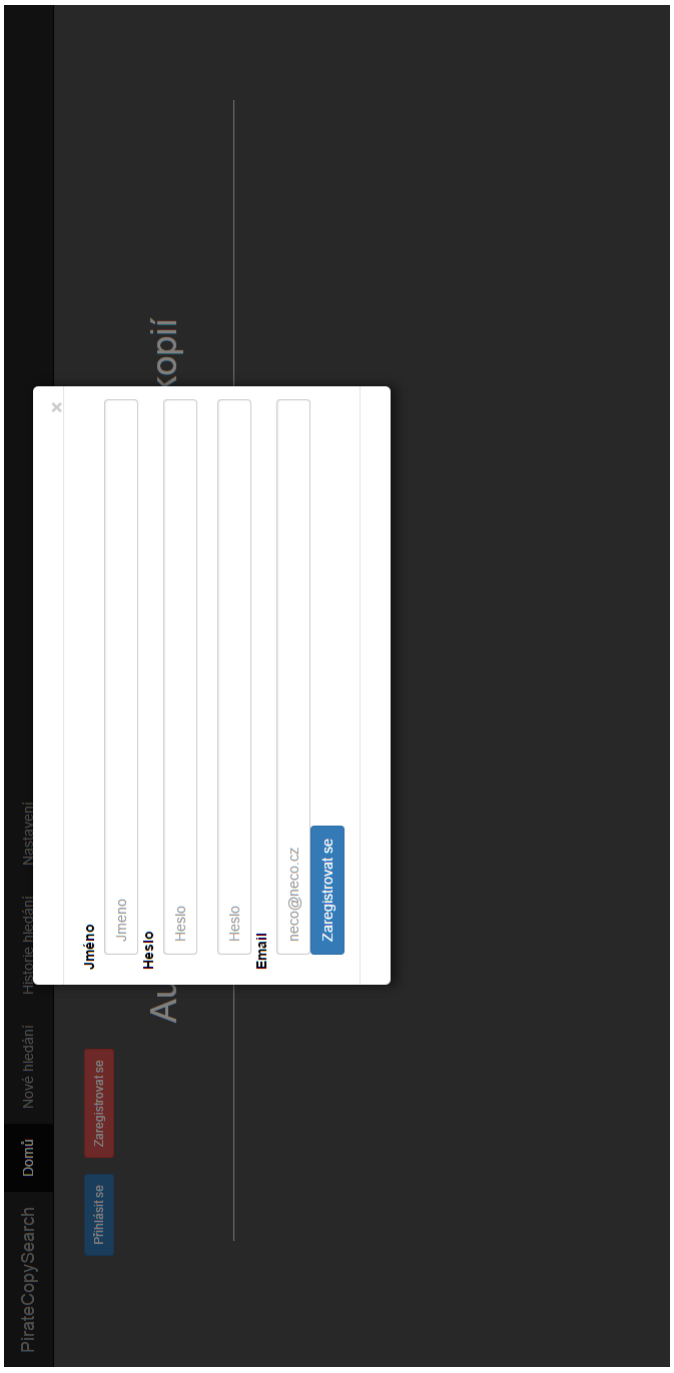
Příloha B

Obrázky

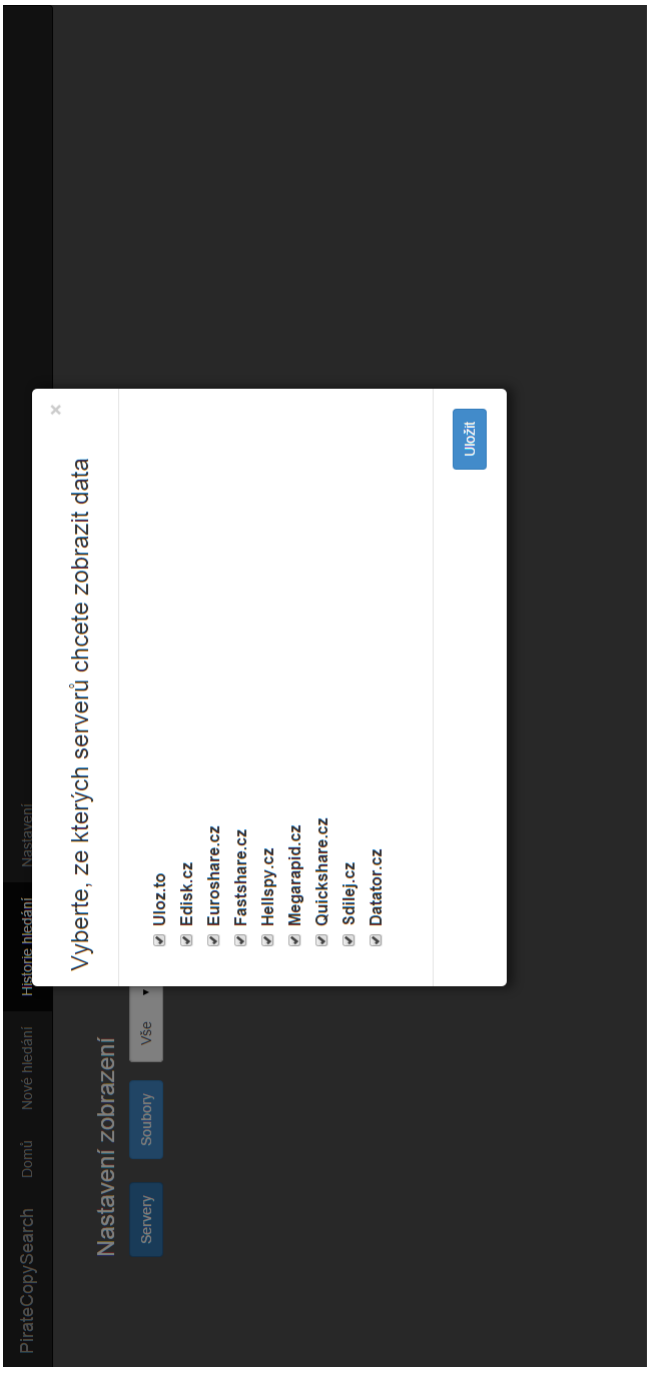


Obrázek B.1: Ukázka responzivního menu

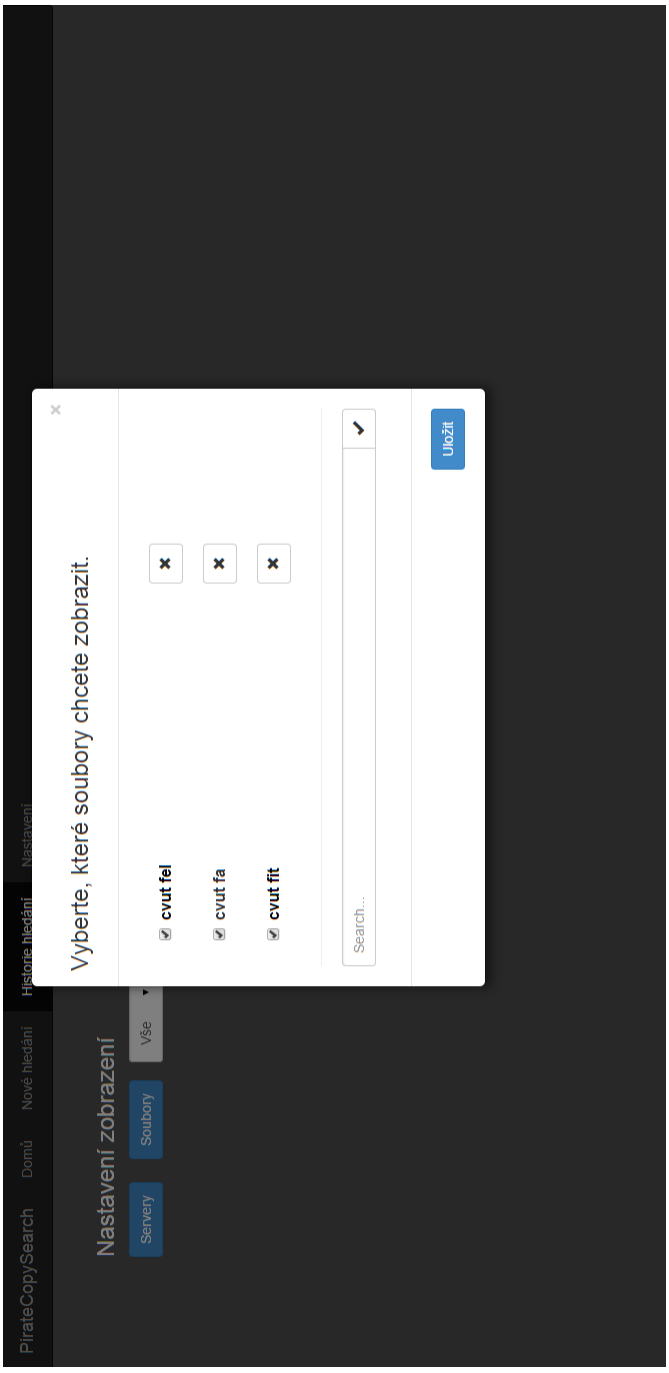
Obrázek B.2: Obrazovka přihlášení



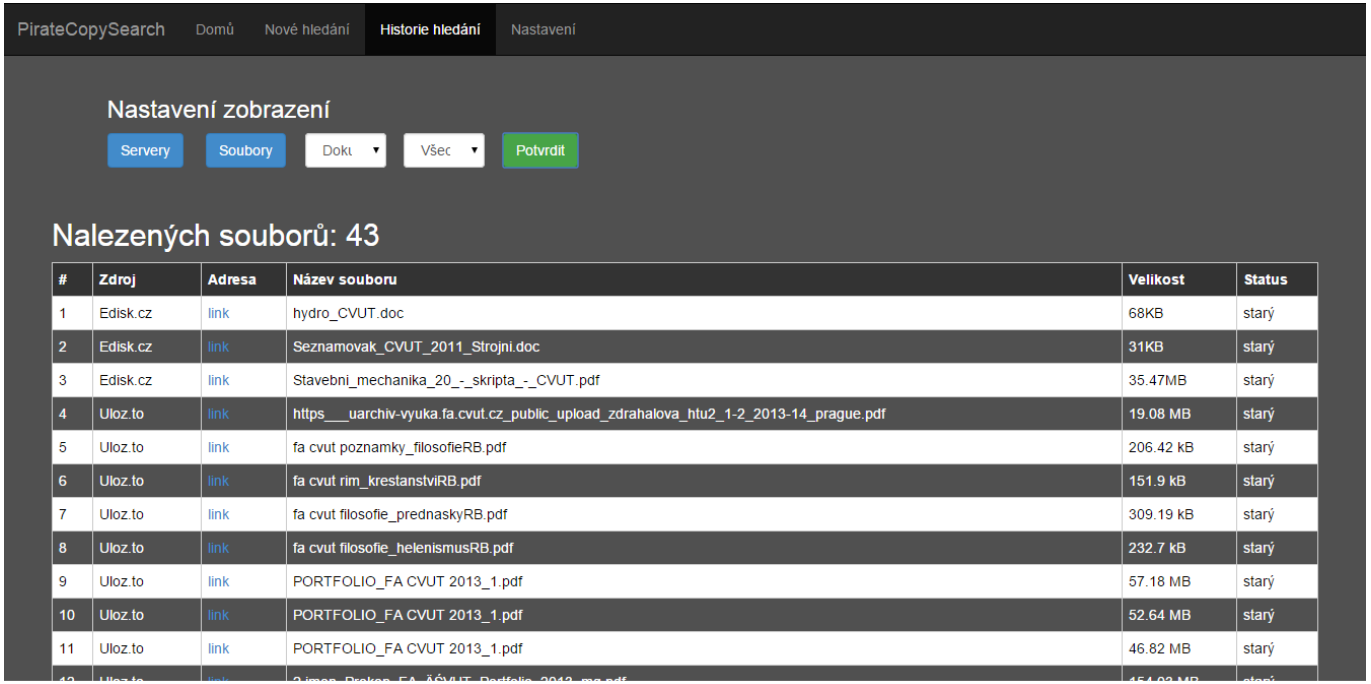
Obrázek B.3: Obrazovka pro registraci



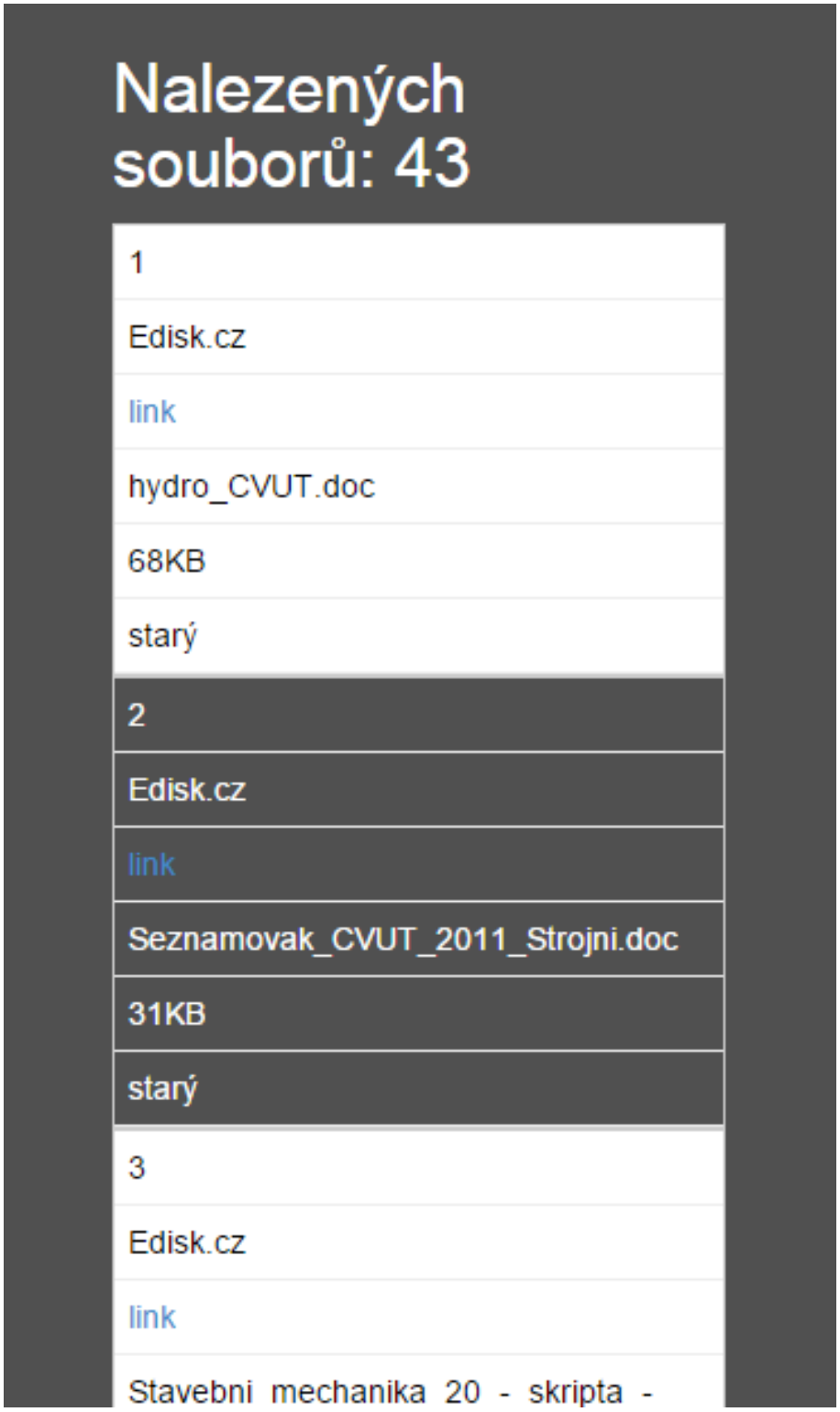
Obrázek B.4: Obrazovka pro výběr serverů



Obrázek B.5: Obrazovka pro výběr souborů



Obrázek B.6: Obrazovka s výsledky hledání



Obrázek B.7: Obrazovka s výsledky hledání 2

Obrázek B.8: Obrazovka nastavení

Příloha C

Skripty na vytvoření databáze

```
CREATE TABLE STAZENY_SOUBOR(
        SID INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,
        nazev VARCHAR2 (50)  NOT NULL ,
        adresa_stazeni VARCHAR2 (150)  NOT NULL,
        velikost varchar2 (10)  NOT NULL,
        typ varchar2 (10)  NOT NULL,
        zobrazeno boolean,
        HID INTEGER NOT NULL,
        SEID INTEGER NOT NULL,
        FOREIGN KEY (HID) REFERENCES HLEDANY_SOUBOR(HID),
        FOREIGN KEY (SEID) REFERENCES SERVER(SEID)
);

CREATE TABLE SERVER(
        SEID INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,
        nazev VARCHAR2 (20)  NOT NULL,
        adresa VARCHAR2 (120),
        trida VARCHAR2 (50)
);
CREATE TABLE HLEDANY_SOUBOR(
        HID INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,
        UID INTEGER NOT NULL,
        nazev VARCHAR2 (20)  NOT NULL,
        FOREIGN KEY (UID) REFERENCES UZIVATEL(UID)
);
CREATE TABLE UZIVATEL(
        UID INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,
        uzivatelske_jmeno VARCHAR2 (20)  NOT NULL,
        email VARCHAR2 (50) NOT NULL,
```

```
        heslo VARCHAR2 (64) NOT NULL,
        frekvence INTEGER,
        info INTEGER
);
```

Příloha D

Obsah přiloženého CD

- `readme.txt` ..... stručný popis obsahu CD
- `exe` ..... adresář se spustitelnou formou implementace
- `src`
  - `impl`
    - `klient` ..... zdrojové kódy klientské aplikace
    - `server` ..... zdrojové kódy serverové aplikace
  - `thesis` ..... zdrojová forma práce ve formátu LaTeX
- `text` ..... text práce
  - `vondrda9_2015.pdf` ..... text práce ve formátu PDF