VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV INFORMAČNÍCH SYSTÉMŮ - DEPARTMENT OF INFORMATION SYSTEMS

WEBOVÝ NÁSTROJ PRO ANALÝZU VELIKOSTÍ ZÁLOH SYSTÉMU BACULA

WEB TOOL FOR ANALYSIS OF THE BACKUP SIZES IN BACULA SYSTEM

BAKALÁŘSKÁ PRÁCE

BACHELOR'S THESIS

AUTOR PRÁCE — FRANTIŠEK DAŇHEL

AUTHOR

VEDOUCÍ PRÁCE — ING. RUDOLF ČEJKA

SUPERVISOR

BRNO 2017

Zadání bakalářské práce/19737/2016/xdanhe04

Vysoké učení technické v Brně - Fakulta informačních technologií

Centrum výpočetní techniky Akademický rok 2016/2017

Zadání bakalářské práce

Řešitel: **Daňhel František**
Obor: Informační technologie
Téma: **Webový nástroj pro analýzu velikostí záloh systému Bacula**
Web Tool for Analysis of the Backup Sizes in Bacula System
Kategorie: Web

Pokyny:

1. Seznamte se se zálohovacím nástrojem Bacula a se způsobem, jak lze z něho získávat informace o provedených zálohách.
2. Navrhněte způsob ukládání sledovaných informací z uskutečněných záloh do databáze.
3. Navrhněte a implementujte webový nástroj, který by zobrazoval vybrané statistiky záloh na webu.
4. Zhodnoťte dosažené výsledky a možnosti dalšího vývoje.

Literatura:

- Dokumentace k Bacule na http://blog.bacula.org/documentation/documentation

Podrobné závazné pokyny pro vypracování bakalářské práce naleznete na adrese http://www.fit.vutbr.cz/info/szz/

Technická zpráva bakalářské práce musí obsahovat formulaci cíle, charakteristiku současného stavu, teoretická a odborná východiska řešených problémů a specifikaci etap (20 až 30% celkového rozsahu technické zprávy).

Student odevzdá v jednom výtisku technickou zprávu a v elektronické podobě zdrojový text technické zprávy, úplnou programovou dokumentaci a zdrojové texty programů. Informace v elektronické podobě budou uloženy na standardním nepřepisovatelném paměťovém médiu (CD-R, DVD-R, apod.), které bude vloženo do písemné zprávy tak, aby nemohlo dojít k jeho ztrátě při běžné manipulaci.

Vedoucí: **Čejka Rudolf, Ing.**, CVT FIT VUT
Datum zadání: 1. listopadu 2016
Datum odevzdání: 17. května 2017

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
Fakulta informačních technologií
Ústav informačních systémů
612 66 Brno, Božetěchova 2

L.S.

doc. Dr. Ing. Dušan Kolář
vedoucí ústavu

Abstrakt

Práce seznamuje s hlavními vlastnostmi zálohovacího nástroje Bacula. Detailněji se zabývá způsobem uložení informací o zálohách v katalogu a možnosti generování informačních e-mailů po provedení jednotlivých záloh. Představuje aktuální stav archivovaných informací o zálohách na Fakultě informačních technologií Vysokého učení technického v Brně. Navrhuje a zabývá se implementací webového nástroje, určeného pro analýzu velikostí, doby trvání a počtu souborů v zálohách. Aplikace zobrazuje sledované veličiny v interaktivních grafech. Jsou použity technologie HTML, CSS, MySQL/MariaDB, PHP, Javascript, jQuery a Flot.

Abstract

The thesis describes the main features of the Bacula backup tool. In more detail, it deals with the method of saving information about backups to Bacula's catalogue and the possibility of generating information emails with details of launched backups. It presents the current state of archived information about backups at the Faculty of Information Technology of Brno University of Technology. It proposes and deals with the implementation of a web tool designed to analyse the size, duration, and number of files in the backups. The application displays tracked quantities in interactive charts. HTML, CSS, MySQL / MariaDB, PHP, Javascript, jQuery and Flot technologies are used.

Klíčová slova

analýza záloh, Bacula, webová aplikace, interaktivní grafy, PHP, Flot

Keywords

analysis of backups, Bacula, web application, interactive charts, PHP, Flot

Citace

DAŇHEL, František. *Webový nástroj pro analýzu velikostí záloh systému Bacula*. Brno, 2017. 31 s. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Čejka Rudolf.

Webový nástroj pro analýzu velikostí záloh systému Bacula

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením Ing. Rudolfa Čejky. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

……...….....

František Daňhel

10. května 2017

Poděkování

Chtěl bych poděkovat mému vedoucímu práce panu Ing. Rudolfu Čejkovi za vedení práce, cenné připomínky, rady a znalosti, které mi v průběhu vypracovávání mé bakalářské práce poskytnul.

Obsah

1 Úvod

Zálohování je činnost, která patří k životu každého systémového administrátora, který má na starosti správu serverů a systémů. Zodpovídá nejenom za správný chod a konfiguraci serverů, ale stará se i o bezpečnost uživatelských dat, které jsou na serverech uloženy. Tato data v dnešní době obvykle mají cenu převyšující cenu samotného hardwarového vybavení, na kterém jsou uložena a využívána. Úkolem je zajistit, aby uživatelé o svá cenná data nepřišli v případě hardwarové chyby zařízení, poškození dat softwarem (ať již úmyslné, či chybou v softwaru), chyby lidského faktoru (ze strany administrátora, či samotného uživatele), fyzickým zásahem neoprávněnou osobou, požárem místnosti apod.

Jakmile je zálohování navrženo a nastaveno, tak přesto, že probíhá po většinu času automatizovaně, neobejde se bez dozoru administrátora/operátora. Je potřeba proces průběžně sledovat a ujišťovat se, že vše probíhá podle plánu a nedochází k abnormálním jevům, jako např. několikanásobný nárůst velikosti záloh mezi dvěma zálohami. Dále je třeba dbát na to, aby veškerá zálohovaná data bylo možné obnovit v případě ztráty dat původních.

Na Fakultě informačních technologií Vysokého učení technického v Brně se používá pro provádění záloh open source produkt Bacula[1]. Jedná se o komplexní a vyspělý nástroj, který si udržuje svůj vlastní katalog se všemi informacemi o provedených zálohách. Prostřednictvím e-mailu je schopen informovat o průběhu a výsledku každé provedené zálohy systémového administrátora.

Cílem této práce je vytvořit webový nástroj, který bude sloužit pro dlouhodobou analýzu dat o provedených zálohách. Nástroj by měl být schopen jednak importovat a analyzovat informace obsažené v elektronické poště rozesílané systémem Bacula a dále by také měl umožňovat průběžné získávání dat obsažených v samotném katalogu zálohovacího řešení Bacula. Webový nástroj by měl být schopen umožnit jednoduché procházení a analýzu informací o provedených zálohách. Data by mělo být možné zobrazit v grafech na základě požadovaných sledovaných parametrů v požadovaném časovém období. Administrátor by měl být jednoduše schopen detekovat krátkodobé anomálie nebo provádět analýzu z dlouhodobého pohledu. Práce využívá reálných informací o provedených zálohách z prostředí FIT, které mají značnou vypovídající hodnotu, a to především díky tomu, že historicky sahají až do roku 2005.

[1] Dostupné na http://blog.bacula.org/

2 Systém Bacula

Na FIT VUT v Brně se používá pro zálohování serverů open source software Bacula. Jedná se o rozsáhlé a flexibilní řešení, které je koncipováno pro zálohování serverů a stanic přes síť na zálohovací server, který provádí zápis na pásky (lze použít i jiné typy médií).

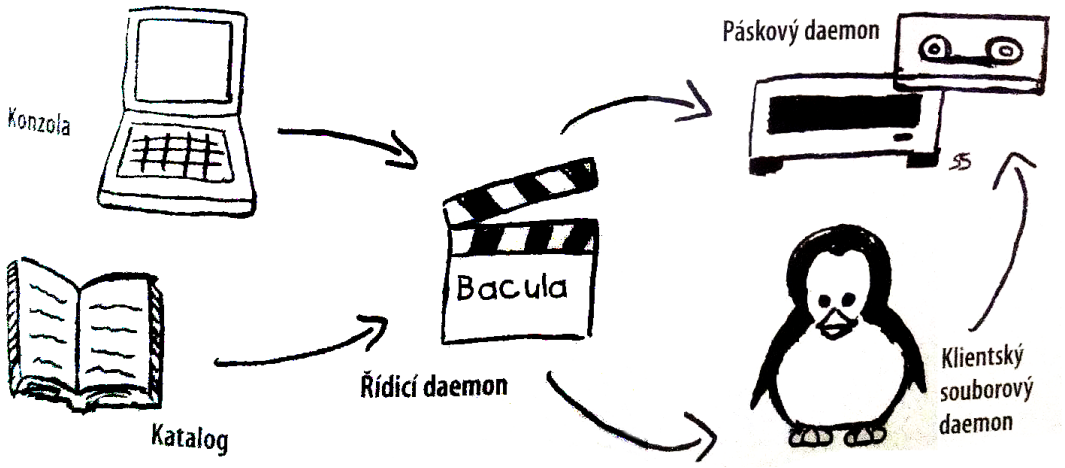
Mezi hlavní výhody Baculy patří především [1]:

- podpora několika různých operačních systémů, včetně Linuxu, Unixu a Windows,
- jako svoji interní databázi může využívat SQLite, MySQL nebo PostgreSQL,
- záloha může být uložena na několika páskách,
- k zálohovaným souborům umí vytvářet jejich MDA5 a SHA1 otisky,
- schopnost verifikovat zálohy,
- podporuje zařízení pro automatickou výměnu pásek.

2.1 Komponenty systému Bacula

Model systému Bacula se skládá z několika komponent, kde každá plní svoji určenou funkci. Tyto komponenty (ve své podstatě se jedná o démony) mezi sebou komunikují pomocí síťového protokolu nad TCP. Je tedy možné provozovat každou komponentu na jiném stroji, ale obvykle se v praxi tento přístup nepoužívá, a některé druhy komponent se provozují společně na stejném stroji.

Každá z komponent obsahuje svůj konfigurační soubor, ve kterém se konfigurují její vlastnosti a chování. U každé komponenty lze definovat TCP port, na kterém má daný démon naslouchat a heslo, které musí znát každá další komponenta, která chce s danou komponentou komunikovat.



Obr. 1 Schéma komponent Baculy, převzato [1 str. 211]

Druhy komponent jsou následující [2]:

Director (řídící démon) zastřešuje všechny operace zálohování, obnovy a verifikace. Je využíván systémovým administrátorem k definování druhu záloh a časovému harmonogramu jejich následného spouštění. Iniciuje a řídí celý proces zálohování a informuje o jeho průběhu.

Console (konzola) je program, který poskytuje administrátorovi připojení a komunikaci s řídícím démonem. V současné době poskytuje jak textové rozhraní příkazové řádky, tak i rozhraní grafické (s využitím knihoven Qt nebo wxWidgets).

File (klientský souborový démon) bývá instalovaný na samostatném stroji, jehož data se budou zálohovat a popř. obnovovat. Program je specifický podle operačního systému daného stroje. Na rozdíl od ostatních komponent je pro tento druh démona typické, že se v síti vyskytuje několikrát (na každém stroji, který se zálohuje). Ostatní komponenty se mohou vyskytnout také na více serverech (a to i včetně řídícího démona), ale nebývá to zvláště obvyklé.

Storage (páskový démon) provádí zapisování a čtení záloh na zálohovací média. Obvykle se jedná o pásky, ale není to podmínkou.

Catalog (katalog) je databáze, do které si Bacula ukládá veškeré informace o provedených i naplánovaných zálohách a obnovách, informace o značení pásek, indexy souborů a jejich umístění na páskách, serverech a další. Protože se práce zabývá analýzou informací právě z této komponenty, bude detailnějšímu popisu katalogu věnována následující podkapitola.

2.2 Katalog Baculy

Bacula nabízí jako volbu úložiště pro katalog následující databáze: MySQL, PostgreSQL nebo SQLite. Volba typu databáze se provádí již při sestavování Baculy, skriptem `./configure`. Např. pro použití databáze MySQL: `./configure -with-mysql`.

V databázi se vytvoří několik tabulek, které Bacula používá k uchování informací o různých úlohách, souborech apod.

2.2.1 Tabulky v databázi

Níže jsou vybrány některé tabulky obsažené v databázi Baculy a je k nim připsán stručný přehled k čemu slouží, popř. co obsahují. Převzato z [3].

Filename, Path – tyto dvě tabulky mají skoro stejnou strukturu. Obsahují sloupec pro číselný identifikátor, a sloupec pro textový řetězec. Jejich účelem je především minimalizovat velikost záznamu v tabulce `File`, kde se odkazuje pouze číselným indexem. Další využití je při vyhledávání jména souboru v zálohách, které se urychlí právě díky tomu, že cesta i název souboru jsou ukládány každý zvlášť.

File představuje jednotlivý soubor v konkrétní záloze. Pokud je tedy tentýž soubor zálohován vícekrát (což běžně je), tak pro něj existuje v tabulce vždy nový záznam. Odkazem na danou zálohu je cizí klíč `JobId`, který se odkazuje na tabulku `Job`. Předpokládá se, že tato tabulka bude v praxi obsahovat nejvíce záznamů. Záznamy se proto postupně promazávají.

Job je nejdůležitější tabulka z pohledu této práce. Obsahuje informace o jednotlivých úlohách, což může být právě konkrétní záloha nebo obnova. S každým záznamem se pojí množství informaci a referencí na ostatní tabulky. Mezi nejzajímavější sloupce z pohledu analýzy uskutečněných záloh patří sloupce: unikátní název úlohy (`Job`); název (`Name`); typ úlohy (`Type`), který nejčastěji nabývá hodnot záloha (`B`) nebo obnova (`R`); úroveň zálohy (`Level`), která může být plná, inkrementální nebo diferenční; reference na klientského souborového démona, který provedl zálohu (`ClientId`); status ukončení úlohy (`JobStatus`), který obvykle nabývá hodnot ze skupin úspěšného ukončení, chybového ukončení, právě běžící a některé další stavy méně obvyklé; časové údaje o naplánování, začátku a ukončení úlohy (`SchedTime`, `StartTime`, `EndTime`); počet zálohovaných (obnovených) souborů, bajtů (`JobFiles`, `JobBytes`) a další reference na sadu souborů, sadu pásek apod.

2.2.2 Promazávání katalogu

Bacula ve svém katalogu informace o provedených zálohách a souborech, které byly jejich součástí, neudržuje navěky. Množství záloh může v průběhu času rychle narůstat. Lineárně vůči uskutečněným zálohám roste i počet záznamů o souborech, protože ke každé záloze se evidují všechny soubory, které jsou její součástí. V případě Fakulty informačních technologií se v každé záloze nachází zhruba v řádu jednotek až desítek milionů souborů. S každou uskutečněnou zálohou se přidá tentýž počet záznamů do databázové tabulky `File`.

Díky tomu, že si Bacula ke každé záloze ukládá soubory v ní obsažené, je možné neobnovovat celou zálohu, ale pouze jednotlivé soubory. Aby se zamezilo tomuto nárůstu katalogu, obsahuje Bacula mechanismus pro mazání těchto záznamů po určité době. K nastavení doby, po kterou se budou informace uchovávat v katalogu, slouží dva parametry, uvedené v konfiguraci řídícího démona (soubor `bacula-dir.conf`) v sekci definující klienta. Jedná se o parametry `File Retention` a `Job Retention`.

```
Client {
    Name = webserver_agata
    Address = 192.168.30.5
    FDPort = 9102
    Catalog = ATE
    Password = "tajne-heslo"
    File Retention = 3 months
    Job Retention = 36 months
    AutoPrune = yes
}
```

Ukázka 1 Ukázka definice klienta v konfiguračním souboru bacula-dir.conf

`File Retention` určuje, jak dlouho se v katalogu mají uchovávat informace o souborech obsažených v záloze. `Job Retention` definuje dobu uchování informace o uskutečněné záloze. Pokud se z katalogu vymažou informace o souborech v záloze (uplatní se `File Retention`), tak ze zálohy sice nelze obnovovat jednotlivé soubory, ale lze stále obnovit celou zálohu a to až do doby, než se pro danou zálohu uplatní `Job Retention`. Jakmile se z katalogu odebere informace o provedené záloze, nelze již zálohu z pásky obnovit (jenom pokud by existoval tzv. Bootstrap soubor). Parametr `AutoPrune` definuje, že se výše definované intervaly obnovy uplatní a záznamy se opravdu smažou. Pokud parametry nejsou definovány, tak implicitní nastavení je 60 dnů pro `File Retention`, 180 dnů pro `Job Rentention` a parametr `AutoPrune` je aktivní [2].

2.3 Rozesílání e-mailových zpráv

V systému Bacula jednotliví démoni generují zprávy o své činnosti a právě zpracovávaných úlohách. Konfigurace předávání zpráv je velmi flexibilní a je možné konfigurovat zvlášť pro každého démona, jaké zprávy bude kam odesílat, popř. zapisovat do logu. Nastavení se provádí v konfigurační sekci `Messages`. Běžně se nastavení koncipuje tak, že klientský souborový démon zprávy odesílá řídícímu démonu a řídící démon, který obdrží zprávy z několika míst současně (např. od páskového i klientského démona) je schopen tyto zprávy shlukovat a poté zasílat např. v jediném e-mailu správci zálohování.

```
Director {
    Name = backup_philip
    Password = "tajne-heslo"
}
FileDaemon {
   Name = webserver_agata
}
Messages {
    Name = Standard
    director = backup_philip = all
}
```

Ukázka 2 Ukázka konfigurace klientského démona pro odesílání zpráv řídícímu démonu

Nastavení na straně řídícího démona probíhá velmi podobně. Můžeme přesně určit, jak se zprávy mají zpracovávat. Mezi přípustné patří výpis zpráv na konzolu, logování do souboru anebo odesílání e-mailem. U odesílání e-mailem se specifikuje příkaz, který má Bacula spustit a jeho případné argumenty, ve kterých lze použít některé zástupné sekvence. Jedna z použitelných sekvencí je např. `%n`, která bude nahrazena názvem úlohy. Jak může vypadat sekce pro konfiguraci zpráv v řídícím démonu, ukazuje Ukázka 3 .

```
Messages {
    Name = Standard
    MailCommand = "/sbin/bsmtp -s \"Bacula: %n\" %r"
    mail = sysadmin@example.com = all
}
```

Ukázka 3 Ukázka konfigurace odesílání zpráv v řídícím démonu

Konkrétní zpráva je programu nastaveném v `MailCommand` předávána na jeho standardní vstup. Může se jednat tedy o jakýkoliv spustitelný soubor/skript, který nemusí odesílat e-maily, ale může provést např. parsování vstupních dat a tato data uložit do databáze. Co se týče konkrétního formátu zprávy, tak ten je generován Baculou a nenašel jsem, zdali je možné tento formát konfigurovat. Na druhou stranu v dokumentaci Baculy [2], je uvedeno *„For additional details, please see the bsmtp - Customizing Your Email Messages section of the Bacula Utility Programs chapter of this manual“*. Bohužel obsažený hypertextový odkaz nefunguje (stránka neexistuje) a zmiňovanou kapitolu jsem v manuálu taktéž nenašel.

3 Návrh webového nástroje

Kapitola se zabývá rozborem aktuálního stavu a existujících nástrojů. Popisuje výhody a nevýhody stávajících řešení a formuluje požadavky na webový nástroj, který je určen k analýze záloh ze systému Bacula.

```
02-Oct 00:05 video3-dir: Start Backup JobId 2048, Job=freebsd.2005-
10-02_00.05.00
02-Oct 00:05 video3-sd: Spooling data ...
02-Oct 00:35 video3-sd: Committing spooled data to Volume.
Despooling 6,777,370,790 bytes ...
02-Oct 00:36 video3-sd: Sending spooled attrs to the Director.
Despooling 92,124,846 bytes ...
02-Oct 00:40 video3-dir: Bacula 1.36.3 (22Apr05): 02-Oct-2005
00:40:06
  JobId:                  2048
  Job:                    freebsd.2005-10-02_00.05.00
  Backup Level:           Full
  Client:                 freebsd-fd
  FileSet:                "freebsd-full" 2005-05-26 18:37:18
  Pool:                   "default"
  Storage:                "overland-localhost"
  Start time:             02-Oct-2005 00:05:02
  End time:               02-Oct-2005 00:40:06
  FD Files Written:       327,961
  SD Files Written:       327,961
  FD Bytes Written:       6,745,812,936
  SD Bytes Written:       6,764,934,336
  Rate:                   3206.2 KB/s
  Software Compression:   None
  Volume name(s):         DAT003L3
  Volume Session Id:      137
  Volume Session Time:    1127479989
  Last Volume Bytes:      374,840,568,856
  Non-fatal FD errors:    0
  SD Errors:              0
  FD termination status:  OK
  SD termination status:  OK
  Termination:            Backup OK
02-Oct 00:40 video3-dir: Begin pruning Jobs.
02-Oct 00:40 video3-dir: Pruned 1 Job for client freebsd-fd from
catalog.
02-Oct 00:40 video3-dir: Begin pruning Files.
02-Oct 00:40 video3-dir: No Files found to prune.
02-Oct 00:40 video3-dir: End auto prune.
```

Ukázka 4 Obsah informačního e-mailu o provedené záloze

3.1 Současný stav

Na Fakultě informačních technologií je v provozu několik desítek serverů, které se pravidelně zálohují. Z databáze Baculy lze získat informace za zhruba 180 dnů (viz kap. 2.2.2). Zároveň

s každou dokončenou zálohou je zaslán administrátorovi informační e-mail, který obsahuje základní informace o proběhnuté záloze, včetně její velikost, stavu provedení apod. Informace za posledních asi 180 dnů lze získat přímo z databáze systému Bacula, kde jsou obsaženy v tabulce `Job` (viz kap. 2.2.1). Starší zálohy lze získávat pouze z informačních e-mailů, které byly obdrženy a byly archivovány.

3.1.1 Formát e-mailu

Ukázka 4 zobrazuje obsah informačního e-mailu, tak jak byl zaslán systémem Bacula. Každý e-mail obsahuje vždy informace vztahující se právě k jedné záloze. E-mail oznamuje úspěšnost, či neúspěšnost provedení dané zálohy a poskytuje podrobné informace. Jedná se o stejné informace, které lze nalézt po provedení zálohy i v databázi Baculy, kdy většina informací je v tabulce `Job`, další informace jako např. název sady souborů nebo úložiště lze nalézt v tabulkách `Fileset`, `Storage`, `JobMedia` apod. E-maily jsou archivovány v obyčejném textovém souboru. V jednom souboru je uloženo několik e-malů. Tento formát je označován jako mbox, e-maily jsou od sebe odděleny prázdným řádkem a nový e-mail se rozpozná vždy na základě detekce řetězce `"From "` (bez uvozovek) na začátku řádku [4].

Zatímco v e-mailu z roku 2005 je obsaženo 24 atributů, v e-mailech novějších jich je 32, což je způsobeno změnou konfigurace v průběhu let a také přechodem na novější verze Baculy. Podstatný rozdíl pro účely importu je v absenci položky `Scheduled time` v prvních několika záznamech. Pokud tedy není tato položka v e-mailu obsažena, tak se tento údaj bere z položky `Job` z data uvedeného za názvem úlohy. Nezáleží na pořadí položek, i když jejich pořadí bylo po celou dobu zachováváno. V e-mailech jsou záznamy různých typů, nejenom zálohy, ale i např. informace o provedených obnovách. V rámci této práce jsou zpracovávány pouze informace o provedených zálohách.

3.1.2 Existující nástroje

V současné době existuje několik webových nástrojů, pomocí kterých lze nastavovat a zobrazovat zálohování v systému Bacula.

3.1.2.1 BWeb

BWeb Management Suite je proprietární produkt firmy Bacula Systems, který je nabízen jako kompletní a jednoduše použitelné webové administrační rozhraní pro Bacula Enterprise Edition. Umožňuje monitorování a administraci jednotlivých úloh. Poskytuje i GUI pro vytváření a správu konfiguračních souborů. Obsahuje několik různých průvodců, které jednoduše provedou administrátora krok za krokem. Také nabízí diagnostické nástroje, pomocí kterých lze zjistit správnost konfigurace databáze, a zdali je systém Bacula správně nainstalován. Převzato z [5].

3.1.2.2 Bacula-Web

Bacula-Web je webový nástroj s otevřeným zdrojovým kódem, který je určen pro monitorování systému Bacula a procházení záznamů v jeho katalogu. Webový nástroj je napsán v PHP a využívá čistě pouze databáze Baculy, odkud získává všechny zobrazované údaje. Nástroj neumožňuje administraci žádných položek, přistupuje ke katalogu pouze pro čtení dat. Umožňuje zobrazit podrobné informace o posledních 150 zálohách. Informace je možné filtrovat podle různých kritérií. Aplikace zobrazuje vybrané veličiny prostřednictvím neinteraktivních grafů. Převzato z [6].

3.1.2.3 Webacula

Webacula je volně dostupná webová administrace s otevřeným zdrojovým kódem, prostřednictvím které lze administrovat většinu nastavení a úloh v systému Bacula. Používá vlastní ACL, pro kontrolu přístupů do webového nástroje. Tyto informace Webacula uchovává ve svých tabulkách, které jsou umístěny v databázi Baculy. Podporuje všechny tři databáze, které podporuje Bacula – MySQL, PostgreSQL i SQLite. Umožňuje informace o zálohách nejen zobrazit, ale i měnit. Lze spouštět konkrétní úlohy, stejně tak i lze spustit obnovu všech nebo vybraných souborů ze zálohy. Také umožňuje připojovat a odpojovat různá úložiště. Převzato z [7].

3.1.2.4 Baculum

Baculum je nástroj, který je součástí systému Bacula verze 7.0 a vyšší. Jedná se o administrační rozhraní, prostřednictvím kterého je možné získávat velké množství informací z katalogu Baculy a zároveň umožňuje spouštět úlohy (ať již zálohu či obnovení), monitorovat stav démonů, podporuje víceuživatelský přístup s konfigurací oprávnění jednotlivých uživatelů, podporuje řízení operací u páskových démonů. Podporuje prostředí s dvěma a více řídícími démony. Umí vytvářet uživatelsky přívětivé grafy a různé hromadné statistiky. Grafy jsou interaktivní a lze přímo ve webovém prohlížeči přibližovat jejich vybrané úseky. Převzato z [8].

Ze zmíněných nástrojů pravděpodobně největší možnosti poskytuje BWeb, který byl vyvinut přímo společností Bacula Systems SA, která stojí i za Bacula Enterprise Edition. Jako jediný z uvedených nástrojů je proprietární, všechny ostatní jsou dostupné zdarma a mají otevřený zdrojový kód (většinou poskytovány s licencí GPL).

Všechny zmíněné aplikace jsou určeny pro administraci systému Bacula. Umožňují většinou konfigurovat a ovládat spouštění záloh (kromě Bacula-Web) a velmi často nabízí možnost generování grafů, pro jednoduchou analýzu základních údajů o zálohách. Z volně dostupných nástrojů je pravděpodobně nejvhodnější volbou, co se týče možností a schopností, Baculum. I přes různé možnosti těchto aplikací, všechny mají jedno společné, a to, že pracují nad aktuálními zálohami, které jsou obsaženy v katalogu Baculy. Žádná z výše uvedených aplikací neposkytuje možnost analýzy starších záloh, které se již v katalogu Baculy nenachází.

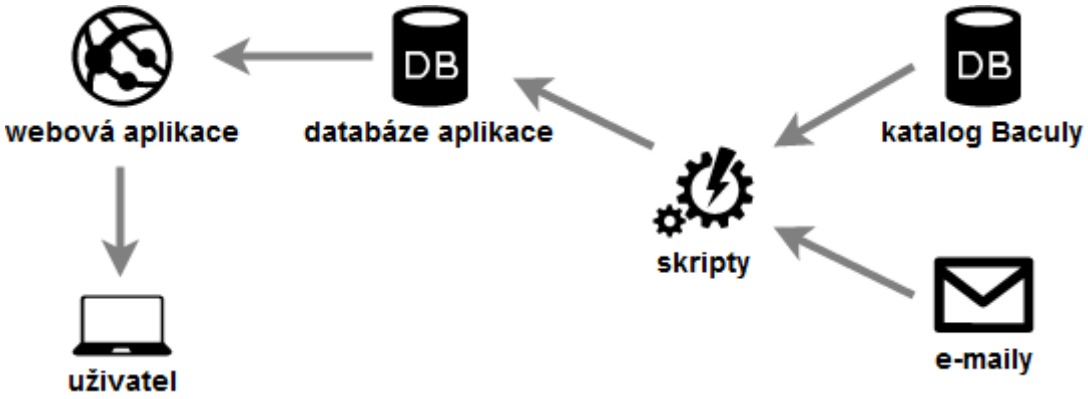
3.2 Specifikace návrhu aplikace

Mělo by se jednat o webovou aplikaci, která bude umožňovat analyzovat zálohy systému Bacula. Nejedná se jen o analýzu informací uložených v katalogu Baculy, protože takové nástroje již existují (viz kap. 3.1.2). Aplikace musí umožňovat importování historických dat, které jsou uchovávány jako uložené informační e-maily obdržené od systému Bacula (více o formátu dat v kap. 3.1.1). Musí být umožněno přidávat do aplikace informace o nových zálohách. Aby se nemusel vždy provádět import přijatých e-mailů, musí být aplikace schopna průběžné automatické synchronizace. Tato synchronizace bude probíhat přímo z katalogem Baculy, odkud lze získat o zálohách mnohem více informací, než z informačních e-mailů, nicméně v rámci zachování konzistence dat v nezávislosti na jejich původci, nebudou ostatní informace z katalogu Baculy brány v potaz.

Mezi zmiňovaná data patří především datum provedení jednotlivých záloh, velikost a počet souborů v záloze obsažených a také doba provádění samostatné zálohy. Všechny tyto informace jsou obsaženy jak v katalogu Baculy, tak i v archivovaných e-mailech.

3.2.1 Architektura aplikace

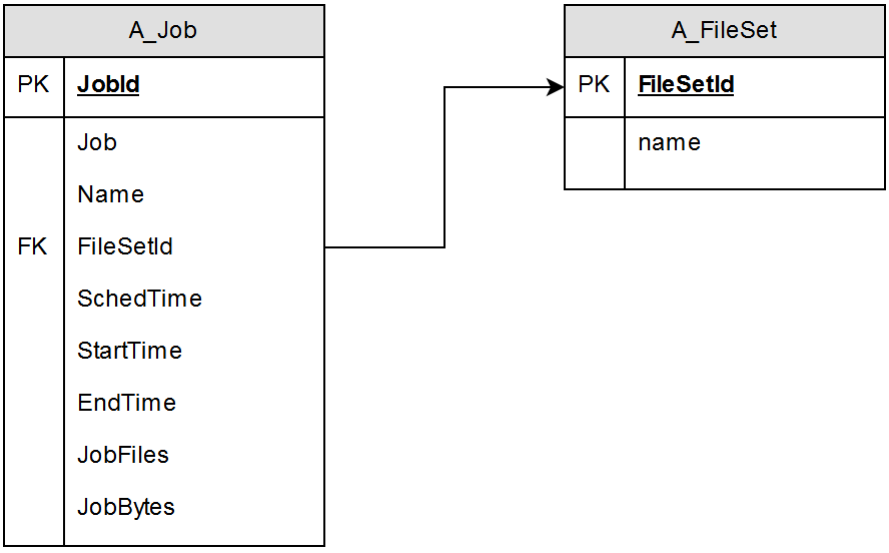
Aplikaci lze rozdělit do několika logických funkčních bloků. Jedno z možných rozdělení je zobrazeno na Obr. 2 . Za jádro aplikace, lze považovat databázi, ve které jsou uložena jednotlivá data o provedených zálohách. Část nazvaná skripty se stará o naplňování dat do databáze aplikace. Jedná se o průběžné automatizované naplňování databáze aplikace z katalogu systému Baculy a také jednorázový manuální import z e-mailů. Dalším logickým blokem je webová aplikace, prostřednictvím které jsou data zobrazována uživateli. Jedná se o část aplikace se kterou bude uživatel nejčastěji pracovat. Šipky na Obr. 2 znázorňují směr toku informací o uskutečněných zálohách.



Obr. 2 Logické bloky aplikace a tok informací o zálohách

3.2.2 Databáze aplikace

Veškerá data, se kterými bude webová aplikace pracovat, budou uloženy v databázi. Obr. 3 znázorňuje navržené tabulky a sloupce v databázi aplikace. Celý návrh se snaží kopírovat strukturu tabulek přímo tak, jak jsou v katalogu Baculy. Oproti katalogu jsou vynechány sloupce, které by zůstaly nevyužity. Tabulky dostaly do svého názvu prefix `"A_"` (bez uvozovek), aby jejich název nekolidoval s názvy tabulek v katalogu Baculy. I přesto však nelze doporučit využití stejné databáze, ale je lepší pro webový nástroj vyhradit vlastní databázi.



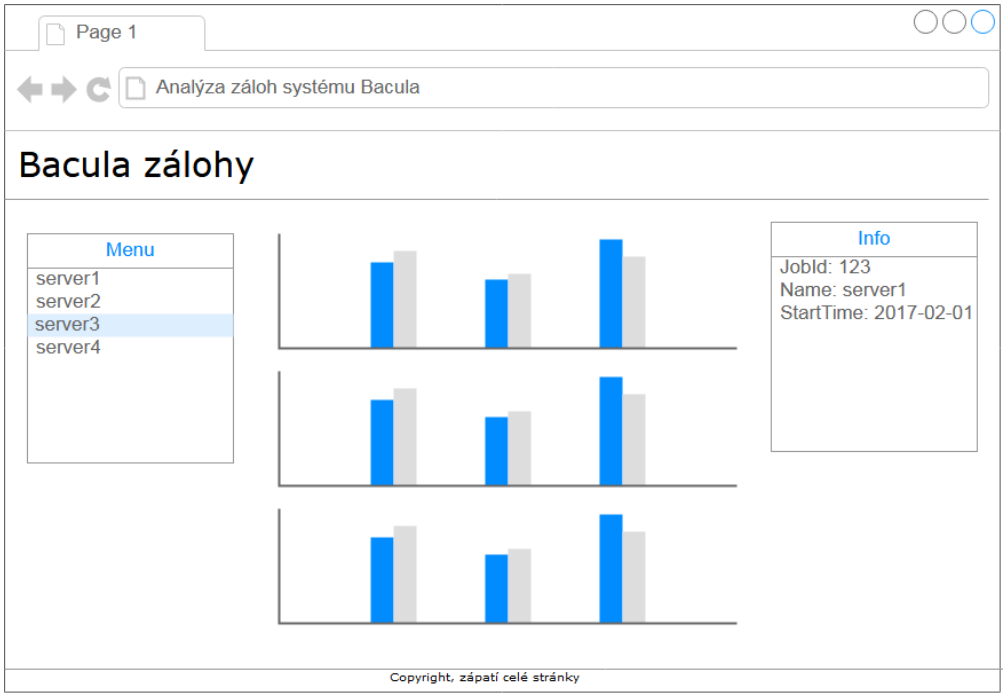
Obr. 3 Návrh tabulek pro databázi aplikace

3.2.3 Skripty

Součástí aplikace budou 2 skripty, které bude možné spouštět z příkazové řádky (CLI). První bude určen pro import e-mailů ze souboru, jedná se o jednorázovou akci, která musí být manuálně realizována uživatelem. Druhý bude mít za úkol synchronizovat katalog Baculy s databází aplikace. Spuštění skriptu pro synchronizaci bude samozřejmě možné také manuálně, nicméně se počítá s automatizací synchronizací za pomocí pravidelného spouštění např. unixovým démonem Cron[2].

3.2.4 Webová aplikace

Bude se jednat o pravděpodobně nejčastěji využívanou část z celého systému. Prostřednictvím webové aplikace bude uživateli umožněno procházet informace o zálohách a provádět jejich analýzu. Aplikace by měla být snadno a jednoduše použitelná. Pro zobrazení dat se bude využívat grafů. Měla by být schopna přizpůsobit se velikosti okna a neměla by mít problém zobrazit se v mobilních zařízeních.



Obr. 4 Návrh GUI webové aplikace

3.3 Použité technologie

Pro samotné skripty a serverovou část webové aplikace bude využito jazyka PHP. Klientská část aplikace bude využívat technologie ve webových prohlížečích, kterými jsou HTML, CSS a Javascript. Pro zobrazování grafů se využije knihovna Flot[3].

Využití jazyka PHP bylo zvoleno z několika důvodů. Jedná se o jednu z nejrozšířenějších platforem na světě, která je provozována na více než třetině všech webových serverů. Objektový model nabízí efektivní tvorbu rozsáhlých projektů, zatímco je možné jej používat i pro jednoduché procedurálně napsané skripty [9 str. 25]. Dalšími důvody bylo, že lze pomocí něho napsat jak skripty

[2] Démon na unixových systémech, který spouští v předem naplánovaných intervalech určené programy.
[3] Oficiální web projektu: http://www.flotcharts.org/

pro příkazovou řádku, tak i serverovou část webové aplikace, díky čemuž je možné sdílet navzájem i společné části kódu. Na Fakultě informačních technologií, je v PHP napsáno několik webových aplikací a vzhledem k tomu, že využití webového nástroje bude probíhat právě v rámci fakulty, tak bude respektována i tato konzistence.

Pro vykreslování grafů v aplikaci je možné využít knihovnu na serverové nebo klientské straně. Vykreslování na serverové straně lze uskutečnit např. za pomocí PHP a jeho rozšíření pro grafickou knihovnu GD [9 str. 322]. Pokud se graf tvoří na serverové straně, dojde k veškerému zpracování dat tam a do prohlížeče se poté stáhne a vykreslí běžný obrázek. Druhou možností je vykreslovat grafy přímo ve webovém prohlížeči. V tom případě se stáhnou ze serveru data, a samotné grafické vykreslení se provádí, nejčastěji za pomocí Javascriptu do HTML elementu `<canvas>`, již na straně webového prohlížeče. Výhodou tohoto řešení je, že se na vykreslený prvek dají navázat různé Javascriptové události a graf je tak pro uživatele interaktivním, což byl hlavní důvod pro zvolení tohoto řešení.

Existuje veliké množství Javascriptových knihoven, umožňující tvorbu grafů. Probíhal výběr z následujících nalezených: Chartist.JS, FusionCharts, Dygraphs, Chart.js, Google Charts a Flot. Při výběru vhodné knihovny byl kladen důraz na: pokud možno, co nejsvobodnější licenci; aby bylo možno vytvářet sloupcové grafy a aby šlo s grafy provádět interaktivní operace jako posun, přiblížení, popř. interakce s jednotlivými vykreslenými body. Na základě kritérií a celkového dojmu, byla zvolena knihovna Flot.

4 Implementace webového nástroje

Tato část se zabývá problematikou samotné implementace aplikace. Objasňuje, jak je aplikace strukturována, popisuje významné třídy a soubory a rozebírá problémy, na které bylo v průběhu implementace naraženo.

4.1 Struktura zdrojových souborů

Ukázka 5 zobrazuje zdrojové kódy aplikace a jejich členění do adresářů. Níže bude stručně vysvětlen popis některých složek a souborů.

```
src
|-- config.ini
|-- scripts
|   |-- create_tables.sql
|   |-- example_mail_data
|   |-- mail_import.php
|   `-- sync_db.php
|-- include
|   |-- Config.php
|   |-- Detail.php
|   |-- FilterJobs.php
|   |-- JobModel.php
|   |-- Job.php
|   `-- JobsParser.php
|-- init.php
`-- www
    |-- css
    |   `-- detail.css
    |-- index.php
    |-- js
    |   |-- BackupModel.js
    |   |-- DetailBox.js
    |   |-- Graphs.js
    |   |-- Layout.js
    |   |-- lib
    |   |   |-- jquery.flot.min.js
    |   |   |-- jquery.flot.navigate.min.js
    |   |   |-- jquery.flot.time.min.js
    |   |   `-- jquery-3.1.1.min.js
    |   |-- main.js
    |   |-- TimeUtils.js
    |   `-- Utils.js
    `-- json.php
```

Ukázka 5 Hierarchie zdrojových souborů aplikace

Konfigurační soubor `/src/config.ini`, umožňuje uživateli aplikace konfigurovat připojení do databází a určovat, ze kterých serverů se mají (popř. nemají) zobrazovat zálohy.

Adresář `/src/scripts/` obsahuje skripty, které slouží pro inicializaci struktury databáze a následně pro získávání dat do databáze. Skript `mail_import.php` slouží pro jednorázový

manuální import a analýzu e-mailů, které jsou uloženy v souboru. Příkladem souboru s e-maily je `example_mail_data`, který obsahuje sadu ukázkových e-mailů, které byly vygenerovány anonymizováním a náhodným zkombinováním reálných dat. Soubor jednak ukazuje, jak vypadají a jakou strukturu mají e-mailové zprávy s reálnými daty a zároveň dává možnost vyzkoušet skript pro import e-mailů a naplnit tak databázi ukázkovými daty. Pro automatizovanou synchronizaci databáze Bacula a databáze aplikace je určen skript `sync_db.php`.

V adresáři `/src/include/` je obsažena definice tříd v jazyku PHP, které využívají skripty i serverová část aplikace. Je dodržena konvence, že každá třída má svůj vlastní soubor (s výjimkou definice potomků třídy `Exceptions`, které jsou umístěny ve stejném souboru, kde je výjimka využívána).

Adresář `/src/www/` obsahuje z většiny klientskou část webové aplikace. Z bezpečnostního hlediska je vhodné, aby webový server byl nastaven pouze pro zpřístupnění právě tohoto adresáře. Je v něm umístěno vše potřebné pro webového klienta. Pokud by nastavení webového serveru směřovalo např. přímo do adresáře `/src/`, tak webová aplikace sice bude také fungovat, ale hrozí riziko úniku přihlašovacích údajů do databází, protože ty jsou uloženy v souboru `/src/config.ini` v podobě nešifrovaného prostého textu. Soubor `/src/www/index.php` představuje vstupní bod do celé webové aplikace, která ke své funkci využívá JavaScriptu[4], který se nachází v `/src/www/js/`. Soubor `/src/www/json.php` zprostředkovává data z databáze ve formátu JSON[5] a také poskytuje jednoduché API pro jejich filtraci.

Adresář `/src/www/js/lib/` obsahuje knihovny třetích stran jQuery a Flot. Tento adresář jako jediný obsahuje převzaté soubory, jichž nejsem autorem. Ve všech těchto souborech je na začátku uvedeno jméno autora a reference na příslušnou licenci, která poskytuje možnost tyto soubory zahrnout, využívat, modifikovat a šířit v rámci celé bakalářské práce.[6]

4.2 Skripty

Součástí práce byla implementace dvou skriptů, které jsou schopny získávat potřebná data a ukládat je do databáze aplikace. Jedná se o soubory, které jsou určeny přímo ke spouštění z prostředí příkazové řádky (také označováno jako CLI).

4.2.1 Synchronizace databází

Skript `sync_db.php` synchronizuje databázi Baculy s aplikační databází. Je navržen, pro pravidelné spouštění např. pomocí démona Cron. Skript naváže 2 databázová spojení do databází, které jsou specifikovány v konfiguračním souboru `config.ini` – v praxi se může jednat o jednu a tu samou databázi, díky tomu, že názvy tabulek jsou navzájem disjunktní. Aby bylo dat přenášených při synchronizaci co nejméně, dotáže se nejprve v databázi aplikace, jaká existuje poslední dostupná záloha, přičemž to, která je poslední se posuzuje na základě databázového sloupce `EndTime`, což označuje čas ukončení provádění konkrétní úlohy. Důležité je, že do aplikační databáze se přenášejí pouze informace o zálohách, které byly již úspěšně dokončeny, zatímco v databázi Baculy jsou i ty zálohy, které teprve proběhnou nebo právě probíhají. Poté se provede dotaz do databáze Baculy a získají se pouze zálohy, které jsou jednak úspěšně dokončené a jejichž

[4] Jedná se o standardizovaný ECMAScript verze 5.1 [10]
[5] Formát dle standardu ECMA-404 [13]
[6] jQuery i Flot jsou šířeny pod licencí MIT

hodnota sloupce `EndTime` je rovna nebo vyšši (jsou novější) než nejvyšší (nejnovější) čas získaný z aplikační databáze.

Mohlo by se zdát, že pro synchronizaci by bylo vhodné využít primární klíč, kterým je v obou tabulkách sloupec `JobId`, obsahující jednoznačný identifikátor úloh. Vzhledem k tomu, že tento identifikátor je vytvářen při definici úlohy, tak jeho posloupnost nemá přímou souvislost s datem spuštění zálohy a tudíž by se nesynchronizovaly všechny úlohy. Jedno z řešení by bylo vždy projít všechny záznamy v aplikační databázi a zapamatovat si všechny použité hodnoty ve sloupci `JobId`. Následně projít všechny záznamy Baculy a přenášet pouze ty, jejichž `JobId` by bylo neznámé. Jedná se sice o řešení možné, nikoliv však efektivní.

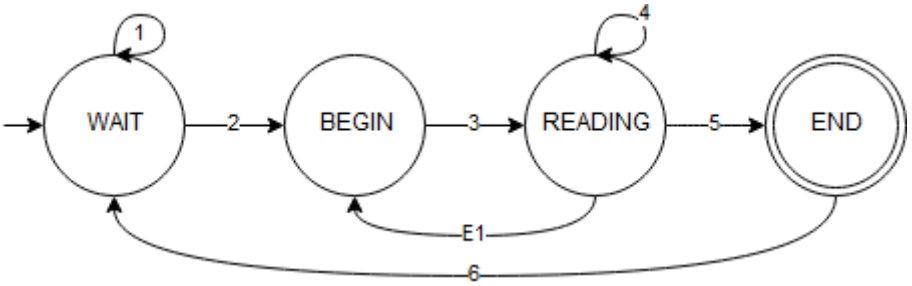
Zvažovaná synchronizace podle nejvyššího času sloupce `StartTime` taky není možná, protože by mohl nastat následující případ: Začne se provádět záloha *A*, po pěti minutách se spustí provádění zálohy *B*. Protože záloha *B* obsahuje málo dat, ukončí se dříve, než záloha *A*. V takovém případě by se záloha *B* přenesla, ale informace o záloze *A* by se již nikdy nesynchronizovala.

Záznam v tabulce obsahuje cízí klíč z tabulky `Fileset`, který je ve sloupci `FilesetId`. Číselnou hodnotu tohoto sloupce ovšem nelze použít a zkopírovat. Hlavním důvodem je skutečnost, že v e-mailech neexistuje žádná položka představující hodnotu sloupce `FilesetId`. Proto synchronizace zkoumá sloupec `Fileset` z tabulky `Fileset` ve kterém je uložen název použité sady souborů. Na základě tohoto názvu se pak získá hodnota primárního klíče `FilesetId` z existujícího záznamu v aplikační databázi, pokud takový záznam neexistuje, vytvoří se nový.

4.2.2 Import e-mailů

Skript `mail_import.php` přijímá jeden volitelný argument, kterým může být název souboru, ve kterém jsou uloženy e-maily. Ukázka 4 zobrazuje část takového souboru. Pokud není žádný argument předán, skript zpracovává standardní vstup. Úkolem skriptu je projít všechny e-maily a rozeznat v nich informace o jednotlivých zálohách, tyto informace následně zpracovat a uložit do databáze.

Vstupní data se postupně čtou po jednotlivých řádcích, které se analyzují. Na jednotlivém řádku se pomocí regulárních výrazů určuje název detekované položky a její hodnota v surovém stavu.



Obr. 5 Stavový automat pro rozpoznávání jednotlivé zálohy

Obr. 5 naznačuje způsob, jakým probíhá detekce jednotlivých záloh. Výchozím stavem je `WAIT`, který reprezentuje čekání na řádek se začátkem zálohy, který je identifikovaný výskytem řetězce `"JobId: "` následovaným číslicí. V tomto okamžiku se přechází po hraně č. 2 a 3 do stavu `READING`, ve kterém jsou rozeznávány všechny další položky o záloze. Jakmile se narazí na řádek obsahující řetězec `"Termination: Backup OK"`, přechází se hranou č. 5 na koncový stav `END`, ve které je ještě záznam validován a standardizován do jednotné podoby. V rámci standardizace se sjednocuje formát data a kontroluje se výskyt všech položek. Ve starších e-mailech se nevyskytuje

položka `ScheduledTime`, proto se u starších e-mailů bere z položky `Name`. Hrana č. E1 označuje přechod v případě, že se v průběhu čtení položek detekuje položka `JobId` (tedy začátek jiné zálohy), aniž by byla předchozí řádně ukončena přechodem do stavu `END`. Takto nekompletní záloha je ignorována.

Jak skript pro synchronizaci databáze, tak skript pro import dat z e-mailů jsou navrženy tak, aby v případě že získaná záloha již existuje, tak se znovu do databáze nepřidává. Je to především z toho důvodu, aby bylo možno provádět import a synchronizaci záloh z několika zdrojů, jejichž zálohy se mohou navzájem částečně nebo úplně překrývat. Je spoléháno na to, že každá záloha má jedinečný identifikátor `JobId` a neexistuje případ, že dvě a více různých záloh by používalo stejnou hodnotu položky `JobId`.

Skript pro synchronizaci databází přidává do databáze aplikace pouze zálohy, které jsou novější než již existující, zatímco skript pro import e-mailů přidává všechny zálohy, které zatím nejsou v databázi uloženy. Proto je při prvotním naplňování databáze daty, vhodnější nejprve spustit skript pro synchronizaci a následně poté až provést import jednotlivých e-mailových souborů.

4.3 Webová aplikace

Tato podkapitola popisuje implementaci jednotlivých částí webového nástroje a zabývá se problémy, které bylo nutné při vývoji řešit.

4.3.1 Serverová část

Na serverové části se využívá skriptovacího jazyka PHP, bez jakýchkoliv speciálních rozšíření nebo frameworků. Úkolem je především vygenerovat webovou stránku s klientskou částí, podle aktuálně zvolené zálohy, zpracovat konfigurační soubor a v neposlední řadě zprostředkovat data klientské části aplikace – webovému prohlížeči.

4.3.1.1 Konfigurační soubor

Konfigurační soubor je umístěn v souboru `config.ini`. Není využíván pouze webovou aplikací, ale zároveň ho používají i skripty určené pro příkazový řádek.

Zpracování konfigurace ve formátu INI je jednoduché především díky své podpoře v PHP prostřednictvím funkce `parse_ini_file`, která zpracuje textový soubor a jeho obsah převede na programové struktury jazyka. Díky jednoduchosti samotného formátu, je i jeho použití pro uživatele snadné.

```
[analytic]
database=bp
user=fdanhel
password=secret23*PaSs

[exclude]
name[]=eva
name[]=adela
```

Ukázka 6 Zkrácený výpis souboru config.ini

Ukázka 6 zhruba znázorňuje, jak vypadá struktura konfiguračního souboru. Názvy sekcí jsou obsaženy v hranatých závorkách. Pod názvem sekce vždy následuje její obsah. Před rovnítkem

je název direktivy, za rovnítkem její hodnota. Pro konfiguraci webového nástroje a skriptů, existují 4 sekce:

- `[bacula]` konfigurace připojení do databáze systému Bacula
- `[analytic]` konfigurace připojení do databáze webového nástroje
- `[exclude]` obsahuje názvy záloh, které se nebudou zobrazovat
- `[include]` obsahuje názvy záloh, které se právě budou zobrazovat

Sekce `bacula` a `analytic` jsou povinné, a musí povinně obsahovat položky `database`, `user` a `password`, které definují připojení do databáze. Volitelně lze využít `host` nebo `socket`. Sekce `exclude` nebo `include` může být použita volitelně a nikdy nelze použít současně obě dvě. Pokud je jedna z těchto sekcí použitá, lze specifikovat jednou nebo více direktivami `name[]` název zálohy. Pokud se použije sekce `include` zobrazí se pouze zálohy, které jsou explicitně vyjmenovány. V případě použití sekce `exclude`, se zobrazí všechny dostupné zálohy, kromě těch, které jsou v této sekci zmíněny.

4.3.1.2 Generování dat ve formátu JSON

Z důvodu potřeby získání dat o zálohách ve formátu JSON, aby se mohla dále zpracovávat v klientské části aplikace, vzniknul webový skript `json.php`. Úkolem skriptu je získat požadovaná data z databáze a tyto záznamy zobrazit ve zmíněném formátu. Specifikace požadovaných dat se provádí prostřednictvím dotazové URL části[7].

Takovýto skript, pak nemusí sloužit jen pro účely této webové aplikace, ale lze ho využít jako API pro jiné nástroje nebo v případě budoucího rozšiřování funkčnosti aplikace. Princip fungování je takový, že se na základě parametrů v URL dynamicky vytvoří odpovídající SQL dotaz, ten se spustí a se získanými daty z databáze se ještě provede transformace, při které se formátují vhodně některé datové typy, pro použití ve výstupním formátu JSON. Pro převod z datových struktur jazyka PHP na formát JSON se používá vestavěná funkce `json_encode`.

Skript nabízí prostřednictvím URL možnost filtrování podle různých kritérií a také různými způsoby. Pokud nejsou zadány žádné parametry, tak jsou vráceny všechny v databázi dostupné zálohy. Ty jsou shlukovány podle jejich jména. Časové údaje jsou předávány způsobem, který se používá v Javascriptu a zároveň ho tak vyžaduje knihovna Flot – číselný typ, udávající časové razítko v milisekundách (počet milisekund od 1. ledna 1970 UTC) [10 str. 165].

Filtrace dat může být ovlivněna specifikací následujících parametrů:

- `names[]` jména záloh, které mají být filtrovány (lze použít vícekrát)
- `filesets[]` jména sad souborů (lze použít vícekrát)
- `level` specifikuje druh zálohy (přípustné hodnoty jsou `full`, `incremental`, `differential`)
- `from` datum spuštění nejstarší zobrazené zálohy (ve formátu `YYYY-MM-DD`)
- `to` datum spuštění nejnovější zobrazené zálohy (ve formátu `YYYY-MM-DD`)
- `last` specifikuje počet dnů do minulosti, ze kterých se mají zobrazit zálohy

Lze specifikovat jeden nebo více ze zmíněných parametrů. Při specifikaci více parametrů musí každá záloha ve výsledné sadě splňovat tyto parametry současně. Malou výjimkou je parametr `last`, který není možné specifikovat společně s `from` a/nebo `to`. Pokud se tak stane, výpis se provede,

[7] část URL za otazníkem označená jako "query" podle RFC 3987 [12 str. 16]

avšak hledí se pouze na parametr `last`, tak jako kdyby `from` a/nebo `to` nebylo součástí URL adresy.

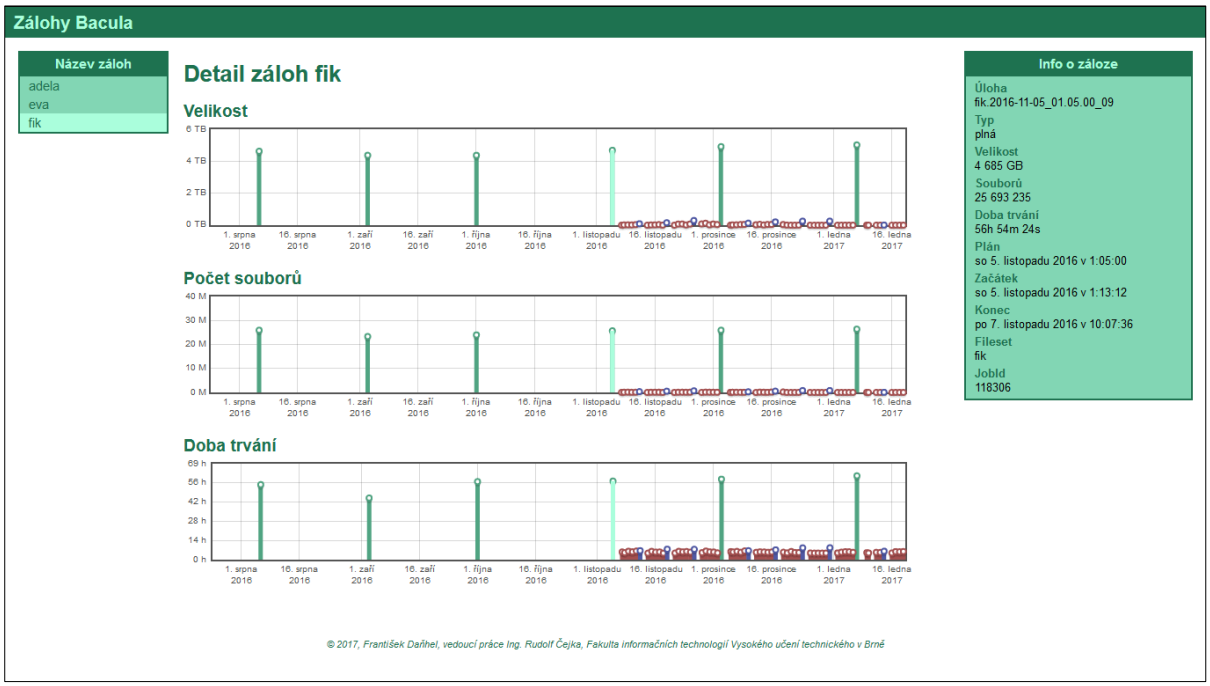
```
SELECT JobId AS id, Job AS job, Name AS name, Level AS level,
FileSet AS fileset, SchedTime AS schedTime, StartTime AS startTime,
EndTime AS endTime, TIMESTAMPDIFF(SECOND, StartTime, EndTime) AS
duration, JobFiles AS jobFiles, JobBytes AS jobBytes FROM A_Job LEFT
JOIN A_FileSet ON A_FileSet.FileSetId = A_Job.FileSetId WHERE 1 AND
Name IN ('merlin') AND StartTime >= '2017-03-01' AND StartTime <=
'2017-03-15 23:59:59' AND Level = 'full' ORDER BY Name,StartTime
```

Ukázka 7 SQL dotaz interně generovaný na základě URL parametrů

Jak konkrétně vypadá interní překlad URL parametrů na SQL dotaz, znázorňuje Ukázka 7 V tomto případě bylo konkrétně použita následující adresa:
`json.php?from=2017-03-01&to=2017-03-15&level=full&names[]=merlin`

4.3.2 Klientská část

Klientskou částí aplikace je interaktivní webová stránka, která se zobrazuje uživateli ve webovém prohlížeči. Webová stránka je sama o sobě generována serverovou částí a to konkrétně skriptem `index.php`. Interaktivní webová stránka nabízí možnost výběru z dostupných záloh, přičemž ty jsou rozlišovány na základě názvu, který ve většině případů identifikuje server, na kterém se zálohy prováděly. Název zálohy se předává prostřednictvím URL parametru `name`. Serverový skript, ještě před vygenerováním webové stránky ověří, zda záloha s daným jménem skutečně existuje. Následně vygeneruje korespondující kostru HTML stránky, do které generuje menu s odkazy na další dostupné zálohy. Zálohy, které se mají v menu zobrazovat lze konfigurovat v konfiguračním souboru (viz kap. 4.3.1.1). V případě, že záloha neexistuje, generuje se pouze neinteraktivní stránka. Pokud dotyčná záloha existuje, do HTML kostry se vygeneruje vložení Javascriptových souborů, které poté převezmou veškerou funkčnost stránky již na straně klienta ve webovém prohlížeči.



Obr. 6 Výsledná webová aplikace pro analýzu záloh

Obr. 6 zobrazuje celkový pohled na webovou aplikaci. V levé části je menu, které zobrazuje dostupné zálohy. Po kliknutí přímo na nadpis menu "Přehled záloh" se celé menu sbalí, popř. rozbalí, čehož lze využít např. na mobilních zařízeních. V prostřední části jsou zobrazeny grafy s konkrétními daty. Grafy zobrazují 3 veličiny: velikost všech souborů v záloze, počet souborů v záloze a doba provádění zálohy. Data v grafech lze interaktivně procházet. Pomocí počítačové myši, lze graf posouvat metodou táhni a pusť (drag&drop), kolečkem myši lze data přibližovat a oddalovat. Zajímavý je princip, kterého se podařilo docílit, a to, že posun nebo přiblížení stačí provádět pouze na jednom ze tří grafů a následně je vše synchronizováno na ostatní grafy, aby byl zachován správný kontext a bylo jasné, že sloupce grafu zobrazené pod sebou reprezentují informace o jedné a té stejné záloze. Po kliknutí na sloupec konkrétní zálohy v grafu (resp. kliknutí na bod na vrcholu tohoto sloupce) se daná záloha zvýrazní napříč všemi grafy a v pravé části stránky se zobrazí detailní informace o zvolené záloze. Všechny prvky jsou stylovány za pomocí CSS takovým způsobem, aby elementy byly přizpůsobivé velikosti obrazovky a aby se zobrazení dokázalo přizpůsobit i mobilním zařízením.

Následující podkapitoly vysvětlují, co se řešilo v rámci implementace klientské části a jak se postupovalo. Pro řešení konkrétního problému, vždy byla vytvořena v Javascriptu třída (resp. prototyp objektu nebo funkce). Je dodrženo opět stejné konvence, jako u serverové části, kdy je každá třída umístěna v samostatném souboru.

4.3.2.1 Model dat

Model dat v klientské části aplikace reprezentuje Javascriptová třída `BackupModel`. Vstupními daty pro konstruktor této třídy je Javascriptová programová struktura, odpovídající jedné konkrétní záloze, která je vrácena serverovým skriptem `json.php`. Data jsou ovšem strukturována mírně odlišně, než je potřeba pro jejich vykreslení do grafu, za pomocí knihovny Flot.

Knihovna Flot požaduje data pro jednotlivé grafy předat ve formě Javascriptového pole, jehož prvkem je pole obsahující dvojici hodnot pro osu X (horizontální) a osu Y (vertikální), např.: `[[1, 1], [2, 4], [3, 9]]`. Dále bylo třeba docílit toho, aby se zálohy jednotlivých úrovní (plná, inkrementální,…) od sebe navzájem barevně odlišovaly. Flot umožňuje nastavit barvu jednotlivým sériím dat, proto bylo potřebné oddělit od sebe zálohy plné, inkrementální a diferenciální. Zároveň je třeba pro každý graf vytvořit vlastní sadu dvojic hodnot osy X, která je vždy časová a osy Y, která se liší v závislosti na typu grafu. Protože sada dat musí být zvlášť pro kombinaci každého typu grafu (velikost záloh, počet souborů a doba provádění zálohy) a každé úrovně zálohy (plná, inkrementální a diferenciální) je třeba vždy celkem generovat 9 sad dat.

Další funkcí modelu dat je možnost vyhledat zálohu podle jejího času spuštění. Této funkce se využívá především v okamžiku, kdy uživatel klikne do grafu a označí konkrétní zálohu. Jednak je třeba získat informace o záloze a zobrazit její detaily a druhým důvodem je, aby bylo možné označit vybranou zálohu i v ostatních grafech − nestačí znát pouze souřadnici X daného bodu, ale je třeba vědět i hodnotu Y u konkrétního bodu, která se vždy liší podle typu grafu.

4.3.2.2 Konfigurace Flotu a interakce s grafy

Všechny grafy jsou vykreslovány pomocí knihovny Flot. Ve třídě `Graphs` je zapouzdřena práce právě se zmíněnou knihovnou. Jedná se především o inicializaci knihovny Flot, která spočívá v nastavení konfiguračních voleb a předání dat pro vykreslování. Data jsou získávána za pomocí objektu, který reprezentuje model dat (viz kap. 4.3.2.1).

Aby bylo možné se v grafu pohybovat interaktivně, využívá se rozšíření Flot Navigate, které umožňuje posun a přibližování grafu pomocí myši. Rozšíření Flot Time umožňuje používat časovou

osu a nabízí různé možnosti formátování a práce s časovými údaji. Nastavují se hranice přiblížení a posuvu grafu. Posun grafu je umožněn půl roku před nejstarší záznam v grafu a půlroku za nejnovější. Přiblížit lze maximálně na celý týden a oddálit lze na časový rozdíl mezi nejstarší a nejmladší položkou grafu zvětšenou o jeden rok. Nastavují se zde také různé série dat a barvy, jakými se budou vykreslovat.

Mezi zajímavou funkci patří synchronizace posuvu a přiblížení grafů. Vzhledem k tomu, že se do grafů vykreslují u každé zálohy vlastnosti, které používají jiné jednotky a jejich vlastnosti se pohybují navzájem většinou v odlišných řádech, nebylo by možné a smysluplné zobrazovat je všechny společně v jednom grafu. Byla by možnost zobrazovat každý graf jen samostatně, ale poté by uživatel ztrácel kontext a nebyla by zřejmé vzájemné souvislosti (např. mezi počtem souborů a velikostí zálohy). Samotná implementace synchronizace pohybu se ve výsledku ukázala být jednodušší, než se na první pohled zdálo. Díky tomu, že plugin Navigate definuje 2 události: `plotzoom` a `plotpan`, je možno navázat na tyto události vlastní funkci, která bude volána vždy, když nastane změna přiblížení nebo posun grafu. Na zmíněné události je navázána metoda `synchronize`, která zjistí aktuální hranice zobrazených dat po posunu nebo přiblížení, a tyto hranice nastaví i u zbývajících dvou grafů.

Dále je zaregistrována metoda `click`, aby se volala vždy při kliknutí do grafu − tedy při události `plotclick`. V těle této metody se provádí zvýraznění zálohy, na kterou bylo kliknuto, ve všech 3 grafech, zároveň se také zobrazí podrobnosti o dané záloze v informačním boxu. K zobrazení detailů o konkrétní záloze se využívá třídy `DetailBox`. Je důležité také připomenout, že knihovna Flot detekuje kliknutí na položku, pouze pokud se klikne přímo nebo do blízkého okolí bodu − nestačí tedy kliknout na sloupec.

Poslední záležitost, kterou tato třída ohledně grafů řeší je spíše estetického rázu. Metoda `resize` dokáže přepočítat a překreslit velikost grafů tak, aby vždy byly široké jako jejich rodičovský element. Využívá se toho především při změně velikosti okna. Zmíněná metoda také řeší jeden drobný problém při vykreslování grafů a tím bylo, že každý graf měl jinak široký na levé straně popisek vertikální osy. Bylo to dáno především rozdílným počtem cifer u čísel a různými jednotkami. Samo o sobě, je toto chování v pořádku, ale při využití navzájem synchronizovaných 3 grafů nevypadalo pěkně, pokud datová oblast každého grafu byla mírně posunutá a jedna konkrétní záloha se nezobrazovala přesně nad sebou. U grafu nelze nastavit šířku popisku osy, ale lze ji programově zjistit. Celý princip pak spočívá v tom, že se nalezne graf s nejširším popiskem vertikální osy. Ostatní grafy se o rozdíl mezi jejich šířkou popisku a tímto maximem posunou doprava nastavením levého vnějšího okraje. Zároveň se o hodnotu posunu zmenší jejich šířka, aby se dosáhlo vzájemného vizuálního zarovnání pravého okraje grafu.

4.3.2.3 Pomocné utility

Aby se docílilo správného formátování údajů jako je datum nebo velikost zálohy, byla vytvořena sada funkcí (statických metod) zapouzdřená třídami `Utils` a `TimeUtils`. Třída `TimeUtils` je primárně určená pro správné generování data v popisku horizontální osy grafu. Flot (resp. jeho rozšíření Time) nabízí několik možností, jak lze nastavit formátování osy. Lze nastavit i vlastní názvy měsíců a dnů, takže by se dalo tímto nastavit český jazyk. Bohužel nastavení formátu se dělá při inicializaci grafu, a v průběhu jej nelze měnit, což má za důsledek, že se datum zobrazuje stále ve stejném formátu (např. měsíc a rok), i když uživatel si graf přibližuje a dostává se do větších detailů. Naštěstí však Flot umožňuje v nastavení grafu přepsat výchozí funkce, které řídí generování popisků osy. Jedná se o dvě funkce: `tickFormatter` a `tickGenerator`. Funkce

`tickGenerator` má za úkol vygenerovat na ose místa, ve kterých se bude vykreslovat popisek. Funkce `tickFormatter` již dostane přímo hodnotu na dané ose a jejím úkolem je pro tuto hodnotu vygenerovat textový řetězec, který se vykreslí do popisku osy. Díky tomu, že funkce `tickFormatter` má přístup i k maximální a minimální hodnotě, která se na ose vyskytuje, je možné uzpůsobit generování popisku nejenom časové hodnotě, ale i stupni přiblížení dat v grafu.

Implementaci funkce `tickGenerator` nebylo nutné přepisovat a je ponechána logika, obsažená v knihovně Flot v rozšíření Time. Využil jsem možnosti definovat si vlastní funkci `tickFormatter`, která generuje data ve vlastním českém formátu a také uzpůsobuje formát data časovému rozsahu zobrazených údajů – např. pouze rok nebo měsíc a rok atd. Díky vlastnímu formátování se rok zobrazuje vždy při všech stupních přiblížení, aby uživatel měl zobrazeno vždy kompletní datum. Původní implementace v knihovně Flot se takto nechovala.

Zatímco třída `TimeUtils` zastává funkci generování popisků časové osy a formátování časových údajů, třída `Utils` obsahuje formátování dalších údajů, jako je časový interval nebo velikost záloh. Většinou se jedná o zvolení vhodného sufixu (jednotky) a popř. vložení mezer pro oddělení řádu. Formátování je mírně závislé na kontextu použití (záleží, jestli se použije pro formátování vertikální osy nebo v detailech o konkrétní záloze).

4.3.2.4 Rozložení prvků a přizpůsobivost

Ve webové aplikaci je pamatováno na různou šířku obrazovky u zařízení, na kterých se aplikace může zobrazovat. Pokud je obrazovka široká, cílem je vyplnit grafy pokud možno co nejvíce místa. V opačném případě, kdy šířka displeje je malá, je třeba položky uspořádat tak, aby na obrazovce bylo vše potřebné. Většiny těchto potřeb se dá docílit za pomocí kaskádových stylů. Existují však výjimky, jako je např. graf knihovny Flot. V tomto případě nestačí napsat správně jen CSS, ale je zapotřebí při změně velikosti okna grafy překreslit tak, aby se jejich šířka přizpůsobila změněné šířce rodičovského elementu.

Šířka menu a detailu zálohy je relativní − je určována na základě použitého písma a konkrétní velikosti obsahu. Šířka těchto dvou elementů je ovšem po vykreslení na daném zařízení již neměnná. Proto je třeba vypočítat šířku prostředního elementu tak, aby vyplnil zbylý prázdný prostor. Je třeba brát zřetel na to, jakým způsobem je definována šířka elementu u prohlížečů podle W3C boxového modelu [11].

5 Testování

V průběhu vývoje aplikace probíhalo průběžné testování, kdy byla ověřována funkce jednotlivých dílčích komponent a později i aplikace jako celku. Testování bylo prováděno manuálně, kdy byla činnost jednotlivých komponent ověřována lidským faktorem.

V první fázi byla ověřována funkce skriptů, které vkládají data do databáze. Testování probíhalo na vybraném vzorku zhruba 10 až 15 vstupních dat, která byla vybírána takovým způsobem, aby pokryla co nejvíce kombinací vstupních faktorů. Při ověřování činnosti parseru (skript `mail_import.php`) probíhalo nejdříve ověření mezi samotným vstupním souborem a obsahem datových struktur v PHP, což bylo prováděno za pomocí funkce `var_dump`, která poskytuje detailní výpis proměnných, a to i těch, které obsahují např. pole nebo objekty. Následně bylo prováděno srovnávání vzorku dat mezi databází aplikace a zdrojovými soubory, popř. zdrojovou databází Baculy.

Při vývoji klientské části, bylo opět na podobném vzorku dat zjišťováno, zdali jsou data správně přenášená do webového prohlížeče. Za tímto účelem bylo využito vývojářských nástrojů, které jsou obvykle obsaženy v každé desktopové verzi známých webových prohlížečů. Obsažená webová konzole umožňuje spouštět Javascriptové příkazy po načtení stránky v reálném čase a díky tomu se dá procházet a zobrazovat obsah jednotlivých proměnných za běhu, popř. lze využít i režimu ladění, při kterém se běh skriptu pozastaví a krokují se jednotlivé řádky kódu. V poslední fázi ověřování bylo kontrolováno, jestli souhlasí údaje v databázi aplikace s tím, jak se vykreslují do grafu a jak se zobrazují v detailu zálohy.

Jakmile byl dokončen vývoj webové aplikace, proběhlo testování webové aplikace v různých prohlížečích na různých operačních systémech (viz Tab. 1). Byla testována především funkčnost uživatelského rozhraní při různých velikostech okna (v případě mobilních zařízení při otočení displeje).

Název OS	**Verze OS**	**Název prohlížeče**	**Verze prohlížeče**
Microsoft Windows	10 Home v. 1607	Microsoft Edge	38
Microsoft Windows	10 Home v. 1607	Mozilla Firefox	53
Microsoft Windows	10 Home v. 1607	Google Chrome	58
Microsoft Windows	7 Professional	Microsoft Explorer	11
Microsoft Windows	7 Professional	Mozilla Firefox	53
Microsoft Windows	7 Professional	Google Chrome	58
Microsoft Windows	7 Professional	Opera	43
openSUSE	Leap 42.2	Mozilla Firefox	52
openSUSE	Leap 42.2	Google Chrome	58
Apple iOS	10.3.1	Safari	10.3.1
Apple iOS	9.3.5	Safari	9.3.5
Google Android	4.4.2	Google Chrome	58

Tab. 1 Přehled prohlížečů, ve kterých byla aplikace testována

Testováním aplikace bylo zjištěno několik chyb a nedostatků, které byly opraveny. Mezi některé z objevených chyb patří např.:

- Špatné zobrazení názvu dnů v detailu zálohy a na časových osách u grafu. Dny byly posunuty z důvodu omylné interpretace číselné hodnoty dne v týdnu, kde 0 odpovídá neděli, nikoliv pondělí [10 str. 167].
- Na mobilních zařízeních bylo nemožné posouvat a přibližovat data v grafu, kvůli absenci myši. Jako alternativní řešení byla přidána tlačítka, která umožňují přiblížení, oddálení a posun dat v grafech.
- Pokud byl prohlížeč (ať už Mozilla Firefox nebo Google Chrome) v celoobrazovkovém módu, probíhal špatně výpočet šířky grafů.

I když se testování snažilo pokrýt určitý počet kombinací vstupních dat a různé webové prohlížeče na různých platformách, tak se jedná pouze o zlomek všech možných kombinací a případů, které mohou nastat. Snahou bylo aplikaci pečlivě otestovat, tak aby se v ní nevyskytovaly žádné chyby, i přesto však nelze zaručit, že nebudou žádné další chyby v budoucnu odhaleny. Stejně tak nelze zajistit i kompatibilitu s budoucími verzemi prohlížečů nebo serverových technologií, které jsou využity.

Obr. 7 Webová aplikace při zobrazení na mobilním zařízení

6 Závěr

Práce měla za cíl vytvořit webový nástroj pro analýzu záloh, uskutečňovaných systémem Bacula. Nejdříve bylo nutné se seznámit se systémem Bacula a nastudovat, jakým způsobem funguje a možnosti jeho konfigurace. Díky tomu, že se jedná o volně dostupný nástroj, tak byla možnost si Baculu stáhnout a vyzkoušet. Nejdůležitější bylo seznámit se s formou, jakou jsou informace o zálohách ukládány v katalogu a jak je možné nastavit generování informačních e-mailů.

Po seznámení se s Baculou, bylo potřebné se seznámit s existujícími e-maily a formou jakou jsou uloženy. Následoval návrh uložení dat do databáze aplikace. Struktura databáze byla navržena s ohledem na již existující schéma v katalogu Baculy. Bylo třeba navrhnout parser, který by uměl zpracovat textové soubory ve formátu mbox, ve kterých byly v jednotlivých zprávách uloženy potřebné informace o jednotlivých zálohách. Pro možnost automatického přidávání nových dat byl navržen způsob jejich synchronizace přímo z katalogu Baculy. Byly implementovány 2 skripty za účelem importu dat o zálohách do databáze aplikace: jeden pro import e-mailů ze souboru mbox, druhý pro synchronizaci databáze aplikace s katalogem Baculy.

Aby byl uživatel schopen procházet a analyzovat data uložená v databázi aplikace, bylo vyvinuto grafické uživatelské rozhraní ve formě webové aplikace. Z technologií na serverové straně bylo využito programovacího jazyka PHP (jak pro serverovou část webové aplikace, tak i pro samostatné skripty spouštěné z příkazové řádky) a databáze MySQL (nebo MariaDB – dle toho, co je v daném systému preferováno). Na straně klientské se využívá technologií, které webové prohlížeče běžně podporují: HTML, CSS a Javascript. Grafické uživatelské rozhraní aplikace zobrazuje zálohy v různých typech grafů, za tímto účelem se využívá knihovny Flot. Je také využito knihovny jQuery, jednak z toho důvodu, že Flot ji pro svoji činnost vyžaduje, ale také z druhého důvodu, kdy poskytuje určitou formu zjednodušení Javascriptového kódu a také nabízí určitou úroveň abstrakce mezi odlišnostmi Javascriptových implementací v běžně používaných webových prohlížečích.

Ačkoliv se, dle mého názoru, v práci podařilo dosáhnout cíle, existuje stále spoustu možností, jak lze v práci pokračovat a dále ji rozvíjet. Proběhlo testování v různých webových prohlížečích na různých zařízeních a operačních systémech. Aplikace byla vyzkoušena, jak cílovými uživateli – administrátory, tak i běžnými uživateli. Pohyb v grafech probíhá pomocí posunu myší a přibližování se provádí kolečkem. Aby bylo možné pracovat interaktivně s grafy i na mobilních zařízeních, byla přidána tlačítka pro ovládání posunu a přiblížení. Interakce s grafy na mobilních zařízeních díky tomu je možná, ale není zcela přirozená. Mnohem vhodnější by bylo umožnit uživatelovi používat standardních gest prsty, která jsou na zařízeních s dotykovou obrazovkou běžná. V pokračování práce tímto směrem, by bylo nutné zvážit, jak moc se aplikace na mobilních zařízeních využívá, popř. by se využívala v případě změny, a také způsob, jakým dané chování implementovat. Je možné, že by bylo nutné vyměnit knihovnu Flot za jinou, popř. pro Flot napsat vlastní rozšíření.

Webová aplikace sice umožňuje uživateli jednoduše analyzovat proběhnuté zálohy, nicméně je na uživateli, aby detekoval případné anomálie. Bylo by velice zajímavé, a pravděpodobně i užitečné rozšířit aplikaci o automatickou analýzu, kdy by výsledná aplikace uměla provádět strojově analýzu dat a v případě detekce nějaké abnormality, by dokázala uživatele upozornit, např. odesláním e-mailu či SMS zprávou. Nemusí se nutně jednat o modifikaci stávající aplikace, ale šlo by na tuto práci navázat, protože webová aplikace zároveň poskytuje webové API rozhraní, prostřednictvím kterého lze strojově získávat informace o zálohách, bez nutnosti přístupu do databáze aplikace.

Literatura

1. **NEMETH, Evi, Garth SNYDER a Trent R. HEIH.** *Linux: kompletní příručka.* 2. aktualizované vydání. Brno : Computer Press, 2008. ISBN 978-80-251-2410-9.
2. **SIBBALD, Kern.** *Bacula Main Reference.* [Online] 2016. [Citace: 20. ledna 2017.] http://www.bacula.org/7.4.x-manuals/en/main/.
3. SIBBALD, Kern. *Bacula Developer's Guide.* [Online] 2016. [Citace: 22. ledna 2017.] http://www.bacula.org/7.4.x-manuals/en/developers.
4. **ROESSLER, Thomas a JANSSEN, Urs.** FreeBSD Man Pages. *mbox(5).* [Online] 19. únor 2002. [Citace: 13. březen 2017.] https://www.freebsd.org/cgi/man.cgi?query=mbox&manpath=FreeBSD+11.0-RELEASE+and+Ports.
5. **BACULA SYSTEMS SA.** *Administration Tools for Bacula Enterprise.* [Online] květen 2014. [Citace: 20. březen 2017.] https://www.baculasystems.com/wp-content/uploads/bacula-enterprise-v8-admintools.pdf.
6. *Bacula-Web.* [Online] [Citace: 15. březen 2017.] http://www.bacula-web.org/.
7. **TIMOFEEV, Yuri.** *Webacula.* [Online] [Citace: 16. březen 2017.] http://webacula.sourceforge.net/.
8. **SIBBALD, Kern.** *Bacula Console and Operators Guide.* [Online] 10. březen 2017. [Citace: 25. březen 2017.] http://www.bacula.org/7.4.x-manuals/en/console/.
9. **GUTMANS, Andi, BAKKEN, Stig Saether a RETHANS, Derick.** *Mistrovství v PHP 5.* Brno : Computer Press, a. s., 2008. 978-80-251-1519-0.
10. **ECMA INTERNATIONAL.** Standard ECMA-262. *ECMAScript Language Specification Edition 5.1.* [Online] červen 2011. [Citace: 5. duben 2017.] http://www.ecma-international.org/ecma-262/5.1/Ecma-262.pdf.
11. **BERT, Bos, a další.** *Cascading Style Sheets Level 2 Revision 1 (CSS 2.1) Specification.* [Online] 7. červen 2011. [Citace: 30. duben 2017.] https://www.w3.org/TR/CSS2/box.html#box-dimensions.
12. **BERNERS-LEE, T., FIELDING, R. a MASINTER, L.** RFC 3986. *Uniform Resource Identifiers (URI): Generic Syntax.* [Online] leden 2005. [Citace: 25. duben 2017.] http://www.rfc-editor.org/info/rfc3986.
13. **ECMA INTERNATIONAL.** Standard ECMA-404. *The JSON Data Interchange Format.* [Online] říjen 2013. [Citace: 5. duben 2017.] https://www.ecma-international.org/publications/files/ECMA-ST/ECMA-404.pdf.

Seznam použitých zkratek

ACL	Access Control List
API	Application Programming Interface
CLI	Command Line Interface
CSS	Cascading Style Sheets
FIT, FIT VUT	Fakulta informačních technologií Vysokého učení technického v Brně
GPL	GNU Public License
GUI	Graphical User Interface
HTML	HyperText Markup Langauge
INI	formát textového konfiguračního souboru
JSON	JavaScript Object Notation
kap.	kapitola
MIT	druh softwarové licence
obr.	obrázek
OS	operační systém
tab.	tabulka
TCP	Transmission Control Protocol
URL	Uniform Resource Locator
UTC	Coordinated Universal Time
W3C	World Wide Web Consorcium

Příloha A: Instalace webové aplikace

Následující postup popisuje instalaci a spuštění aplikace. Postup byl otestován v prostředí serveru SEC6NET, kde je použit operační systém FreeBSD 11, nicméně měl by podobně fungovat i v jiných prostředích.

Manuál počítá s tím, že uživatel již nainstaloval databázi MySQL nebo MariaDB, PHP a webový server (např, Apache HTTPD) a že je toto prostředí správně nakonfigurováno pro standardní běžné použití. Aplikace nevyužívá žádné speciální konstrukce, takže by neměl být problém spustit vše na běžně používaných stabilních verzích programů. Tab. 2 zobrazuje konfigurace, na kterých byla otestována funkčnost serverové části aplikace.

Operační systém	**Verze PHP**	**Databáze**
FreeBSD 11	5.6.30	MySQL 5.6.30
openSUSE Leap 42.2	7.0.7	MariaDB 10.0.27
CentOS 7	5.4.16	MariaDB 5.5.52

Tab. 2 Otestované serverové konfigurace

Na počítač, kde chceme spustit serverovou část aplikace, je nutné překopírovat obsah složky `/src/` na přiloženém CD. Provedeme vytvoření tabulek v databázi. K tomuto účelu stačí provést obsah SQL skriptu `/src/scripts/create_tables.sql`. Lze tak učinit např. příkazem:

```
mysql –u uzivatel –p –D databaze < create_tables.sql
```

Nyní je potřeba v souboru `/src/config.ini` upravit hodnoty pro připojení do databáze. V sekci `[analytic]` se konfigurují údaje pro připojení do databáze aplikace, v sekci `[bacula]` pak údaje pro připojení do databáze systému Bacula (využíváno k synchronizaci). Po správném nastavení připojení do databáze lze provést import dat ze souboru s e-maily. Skript pro import, stejně jako ukázková data se nachází ve složce `/src/scripts/` (reálná data nejsou v rámci práce zveřejňována, lze je poskytnout na vyžádání). Provedení importu e-mailů:

```
php mail_import.php example_mail_data
```

Následně je třeba nakonfigurovat webový server tak, aby směřoval do složky `/src/www/`. Není vhodné zveřejňovat prostřednictvím webového serveru celou složku `/src/`, protože např. soubor `config.ini` obsahuje heslo pro připojení do databáze.

Pokud nelze konfigurovat webový server, tak se doporučuje přesunout složku `/src/www/` jinam do cesty webového serveru a následně upravit v souborech `index.php` a `json.php` na prvních řádcích cestu k souboru `init.php`.

Pro konfiguraci synchronizace je třeba nastavit démona CRON, aby spouštěl v požadovaných časových intervalech skript `/src/scripts/sync_db.php`.