

Identifikace vlastníků bitcoinových adres

Identification of Bitcoin Address Owners

Bc. Adam Šárek

Diplomová práce

Vedoucí práce: Ing. Jan Plucar, Ph.D.

Ostrava, 2023

Zadání diplomové práce

Student: **Bc. Adam Šárek**

Studijní program: N2647 Informační a komunikační technologie

Studijní obor: 2612T025 Informatika a výpočetní technika

Téma: Identifikace vlastníků bitcoinových adres
Identification of Bitcoin Address Owners

Jazyk vypracování: čeština

Zásady pro vypracování:

Bitcoin je v současnosti nejrozšířenější kryptoměnou na planetě. Pomocí kryptoměny Bitcoin je realizován decentralizovaný elektronický platební systém, jehož základy definoval člověk nebo skupina lidí používajících alias Satoshi Nakamoto. Pro identifikaci odesílatele a příjemce platby se využívá Bitcoinových adres. Bitcoinová adresa je jedinečný identifikátor, který slouží jako virtuální místo, kam lze kryptoměnu odeslat.

V této práci se student zaměří na tvorbu prostředí a nástrojů pro deanonymizaci vlastníků Bitcoinových adres. Proces deanonymizace je proces propojení veřejné bitcoinové adresy s digitálním identifikátorem uživatele (přezdívka, email a v omezených případech i jméno a příjmení). Navržené prostředí bude obsahovat databázi Bitcoinových adres, u kterých budou doplněny veškeré nalezené informace. Dále musí student vytvořit prostředí (např. webovou stránku), ve kterém bude schopen uživatel data procházet a ručně editovat. Posledním úkolem je návrh a implementace nástroje, který by byl schopen automatizovaně vyhledávat, analyzovat a ukládat informace o Bitcoinových adresách.

Hlavní body zadání:

1. Rešerše aktuálního stavu.
2. Návrh prostředí, které se bude skládat z několika samostatných nástrojů a databáze Bitcoinových adres.
3. Implementace nástrojů.
4. Otestování řešení a diskuze výsledků.

Seznam doporučené odborné literatury:

1. WU, Yan, et al. A bitcoin transaction network analytic method for future blockchain forensic investigation. IEEE Transactions on Network Science and Engineering, 2020, 8.2: 1230-1241.
2. HONG, Seongho; KIM, Heeyoul. Analysis of Bitcoin exchange using relationship of transactions and addresses. In: 2019 21st International Conference on Advanced Communication Technology (ICACT). IEEE, 2019. p. 67-70.
3. FLEDER, Michael; KESTER, Michael S.; PILLAI, Sudeep. Bitcoin transaction graph analysis. arXiv preprint arXiv:1502.01657, 2015.
4. MARCHENKO, Yuriy; KNOTTENBELT, William J.; WOLTER, Katinka. EthExplorer: A Tool for Forensic Analysis of the Ethereum Blockchain. In: European Workshop on Performance Engineering. Springer, Cham, 2019. p. 100-117.

Formální náležitosti a rozsah diplomové práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí diplomové práce: **Ing. Jan Plucar, Ph.D.**

Datum zadání: 01.09.2022

Datum odevzdání: 30.04.2023

Garant studijního oboru: prof. RNDr. Václav Snášel, CSc.

V IS EDISON zadáno: 07.11.2022 11:59:22

Abstrakt

Cílem této práce je nejprve prozkoumat existující nástroje a databáze obsahující Bitcoinové adresy včetně dalších údajů s nimi spojených a poté implementovat vlastní řešení. Toto řešení se skládá ze dvou nástrojů z nichž jeden na pozadí prochází a stahuje zdrojový kód z webových stránek vybraných nástrojů a sjednocuje data na nich obsažené do jedné databáze. Druhým nástrojem je pak web, který data zpřístupňuje uživateli na základě jeho role v systému. Uživatel má možnost vyhledat konkrétní kryptoměnovou adresu či získat seznam adres, ve kterém lze filtrovat dle kryptoměny či zdroje dat. Tento nástroj pak navíc obsahuje API, která nabízí stejnou funkcionalitu bez nutnosti použití webového rozhraní. Nástroj pro sběr dat je navržen v jazyce Python s využitím PostgreSQL databáze. Web je napsán v JavaScriptu v prostředí Node.js. Oba nástroje je možné provozovat lokálně či na serveru.

Klíčová slova

Bitcoin; Bitcoinová adresa; pseudonymní; databáze; nahlášení; crawler; web; API; Python; Node.js; JavaScript; PostgreSQL

Abstract

The goal of this work is to first examine existing tools and databases containing Bitcoin addresses including other data associated with them, and then implement a custom solution. This solution consists of two tools, one of which in the background goes through and downloads the source code from the web pages of the selected tools and unifies the data contained on them into one database. The second tool is a website that makes data available to users based on their role in the system. The user can search for a specific cryptocurrency address or get a list of addresses that can be filtered by cryptocurrency or data source. This tool also contains an API that offers the same functionality without the need to use a web interface. The data collection tool is designed in Python using the PostgreSQL database. The website is written in JavaScript in the Node.js environment. Both tools can be run locally or on the server.

Keywords

Bitcoin; Bitcoin address; pseudonymous; database; report; crawler; website; API; Python; Node.js; JavaScript; PostgreSQL

Obsah

| | |
|---|-----------|
| Seznam použitých symbolů a zkratek | 7 |
| Seznam obrázků | 9 |
| Seznam tabulek | 10 |
| 1 Úvod | 11 |
| 2 State of the art | 13 |
| 2.1 Práce shromažďující BTC adresy za účelem odhalení podvodů | 13 |
| 2.2 Databáze adres a nahlášených podvodů | 17 |
| 2.3 Shrnutí relevantních zdrojů | 19 |
| 3 Analýza požadavků | 20 |
| 3.1 Požadavky | 20 |
| 3.2 Případy užití | 21 |
| 3.3 Architektura | 23 |
| 3.4 Návrh databáze | 24 |
| 4 Implementace crawleru | 30 |
| 4.1 Správa databáze | 31 |
| 4.2 Spuštění jako správce | 33 |
| 4.3 Procházení zdrojů | 33 |
| 4.4 Propojování dat s adresami | 38 |
| 4.5 Zjištění kryptoměny adresy | 38 |
| 4.6 Automatizovaný běh | 39 |
| 5 Implementace webu | 40 |
| 5.1 Vyhledávání kryptoměnové adresy | 41 |
| 5.2 Seznam adres | 49 |
| 5.3 Statistiky | 50 |

| | | |
|----------|-----------------------------|-----------|
| 5.4 | Uživatelské účty | 51 |
| 5.5 | API | 55 |
| 6 | Použité technologie | 57 |
| 6.1 | PostgreSQL | 57 |
| 6.2 | Node.js | 58 |
| 6.3 | Express.js | 58 |
| 6.4 | EJS | 58 |
| 6.5 | GitHub | 59 |
| 7 | Závěr | 60 |
| | Literatura | 62 |
| | Přílohy | 63 |
| A | Návod k použití | 64 |
| A.1 | Příprava nástrojů | 64 |
| A.2 | Spuštění nástrojů | 65 |

Seznam použitých zkratek a symbolů

| | |
|---------|---|
| BTC | – Bitcoin |
| altcoin | – Alternativní kryptoměna, označení pro kryptoměny mimo Bitcoin |
| SQL | – Structured Query Language |
| PK | – Primary key |
| FK | – Foreign key |
| UQ | – Unique key |
| WAL | – Write-Ahead Log |
| HTML | – Hyper Text Markup Language |
| CSS | – Cascading Style Sheets |
| JS | – JavaScript |
| EJS | – Embedded JavaScript templates |
| JSON | – JavaScript Object Notation |
| CSV | – Comma-separated values |
| XML | – Extensible Markup Language |
| TXT | – Text file |
| GZ | – GNU zip |
| GNU | – GNU's Not Unix! |
| PHP | – PHP: Hypertext Preprocessor |
| UML | – Unified Modeling Language |
| UC | – Use Case |
| ID | – Identification |
| SW | – Software |
| OS | – Operating system |
| UAC | – User Account Control |
| JIT | – Just-in-Time |
| LTS | – Long-term support |
| API | – Application Programming Interface |
| URL | – Uniform Resource Locator |

| | |
|-------|--------------------------------------|
| IP | – Internet Protocol |
| HTTP | – Hypertext Transfer Protocol |
| HTTPS | – Hypertext Transfer Protocol Secure |
| RAM | – Random Access Memory |
| USA | – United States of America |

Seznam obrázků

| | | |
|-----|--|----|
| 2.1 | Příklad Bitcoinového generátoru [1] | 14 |
| 2.2 | Proces shromažďování dat o politických kandidátech USA [4] | 15 |
| 3.1 | Diagram případů užití | 22 |
| 3.2 | Diagram komponent systému | 23 |
| 3.3 | Diagram vztahu entit databáze | 24 |
| 5.1 | Příklad nahlášení obsahujícího jméno | 46 |
| 5.2 | Příklad nahlášení obsahujícího přezdívkou | 47 |
| 5.3 | Navigace u stránkování obsahu | 48 |
| 5.4 | Ukázka části seznamu BTC adres | 49 |
| 5.5 | Ukázka části statistik | 50 |
| 5.6 | Registrační a přihlašovací formulář | 52 |
| 5.7 | Správa uživatelů | 54 |

Seznam tabulek

| | | |
|------|---|----|
| 2.1 | Zdroje kryptoměnových adres | 19 |
| 3.1 | Požadavky pro uživatelské účty | 20 |
| 3.2 | Tabulka account | 25 |
| 3.3 | Tabulka address | 25 |
| 3.4 | Tabulka address_data | 26 |
| 3.5 | Tabulka currency | 26 |
| 3.6 | Tabulka data | 27 |
| 3.7 | Tabulka role | 27 |
| 3.8 | Tabulka session | 27 |
| 3.9 | Tabulka source | 28 |
| 3.10 | Tabulka source_label | 28 |
| 3.11 | Tabulka source_label_url | 29 |
| 3.12 | Tabulka token | 29 |
| 3.13 | Tabulka url | 29 |
| 5.1 | Typy informací obsažené v nahlášení kryptoměnové adresy | 44 |
| 5.2 | Možnosti jednotlivých uživatelských rolí | 53 |
| 5.3 | Limity počtu požadavků na API pro jednotlivé uživatelské role | 53 |
| 5.4 | Seznam funkcí API | 55 |
| 5.5 | Seznam parametrů v požadavcích API | 56 |

Kapitola 1

Úvod

Motivací práce bylo navrhnout a implementovat prostředí složené z několika nástrojů s cílem vytvořit a udržovat databázi Bitcoinových adres. Databáze kromě Bitcoinových adres obsahuje také další informace jako uživatelské nahlášení podvodů spojených s adresami a zdroje, ze kterých byly nahlášení získány. Tyto informace jsou v databázi vzájemně propojeny, což umožňuje jejich nalezení po vyhledání určité adresy.

Pro uživatele je primárním cílem vyhledávání získat informace o možném riziku spojeném s danou adresou. Výhodou agregace nahlášení z několika zdrojů je to, že u vyhledané adresy lze všechny zobrazit přehledně na jednom místě bez nutnosti navštívit jakýkoliv externí web. Následnou analýzu pak umožňují odkazy na přesné místo, kde byly tyto informace získány a také odkazy na blockchainové průzkumníky, kde je možné nalézt veškeré transakce dané adresy.

Databáze je přístupná na webu či skrze API a je možné v ní vyhledávat konkrétní adresy či procházet seznam nahlášených adres, ve kterém lze filtrovat adresy dle zdroje, na kterém byla adresa nahlášena či dle kryptoměny. Volba kryptoměny je zde proto, jelikož v rámci průzkumu stávajících řešení byly nalezeny zdroje, které z většiny obsahují i jiné než jen Bitcoinové adresy. Díky tomu tak byla výsledná databáze rozšířena i o nahlášení jiných kryptoměn a to z toho důvodu, že tak poskytuje širší využití. Stále je však databáze zaměřena primárně na Bitcoinové adresy a je možné v případě potřeby toto rozšíření později odebrat.

Co se týká obsahu jednotlivých kapitol, tak první kapitola se věnuje stávajícímu stavu nástrojů, které shromažďují Bitcoinové adresy jejich vyhledáváním na internetu. Cílem této kapitoly je nejen získat přehled o možných řešeních, ale také ujasnit si, jak by mohla finální databáze vypadat. Jsou zde také uvedeny zdroje, které byly dokonce použity při následné implementaci řešení.

Další kapitola se pak zabývá analýzou požadavků, případy užití jednotlivých nástrojů, architekturou systému a návrhem databáze. Tyto informace jsou doplněny o vhodné UML diagramy.

Jelikož se řešení skládá z několika nástrojů, tak byla jejich implementace rozdělena do dvou samostatných kapitol.

První z nich se zabývá crawlerem. Crawler nejprve vytváří databázi, zavádí určité optimalizace databázového serveru a poté prochází požadované zdroje. Procházení začíná získáním seznamu všech Bitcoinových adres z blockchainu. Následně jsou procházeny zdroje obsahující seznamy nahlášení a zdroje které je možné prohledávat pouze pomocí jejich vyhledávače. K tomu jsou také mimo jiné potřeba již získané adresy z blockchainu. Data jsou po stažení propojena s adresami, aby bylo možné je použít při vyhledávání a nakonec jsou také identifikovány kryptoměny adres, kterým by tato informace v databázi chyběla. Celkově je v rámci implementace crawleru zavedeno několik optimalizací, aby byl celý proces co možná nejrychlejší, nicméně i tak je jeden běh programu otázkou víceméně několika dnů. To je také jedním z důvodů, proč je celý proces co možná nejvíce automatizován, aby po spuštění nástroje nebylo nutné do programu jakkoliv zasahovat.

Webový nástroj se v podstatě skládá ze 2 nástrojů, jelikož kromě webové aplikace je jeho součástí také API. Implementace webu nejprve představuje řešení vyhledávání adresy a to, jaké informace jsou u dané adresy na webu obsaženy. Dále se zabývá deanonymizací vlastníka Bitcoinové adresy pomocí těchto informací a jejich důvěryhodností. Následně se věnuje stránce se seznamem nahlášených adres a způsobu jejich filtrování. Zmíněná je také stránka se celkovými statistikami aktuálního stavu databáze. Dále pak přibližuje zapojení uživatelských účtů a jejich možnosti nejen v rámci webové aplikace, ale také v rámci API. Jaké funkce API nabízí a pro jaké účely byla vytvořena, představuje poslední část implementace.

V poslední kapitole se práce zabývá popisem jednotlivých technologií použitých při implementaci. Jsou zde zmíněné technologie PostgreSQL, Node.js, Express.js, EJS a GitHub.

Součástí práce je také příloha zabývající se přípravou a spuštěním obou nástrojů. Nástroje je možné zprovoznit lokálně na zařízení s operačním systémem Windows 10 či novějším.

Kapitola 2

State of the art

2.1 Práce shromažďující BTC adresy za účelem odhalení podvodů

2.1.1 Automatická detekce a analýza podvodu Bitcoinového generátoru

Tato práce pojednává nástroji sloužícímu k vyšetřování tzv. „podvodu Bitcoinového generátoru“. Ten funguje na jednoduchém principu, kdy podvodník se s pomocí jednoduchého webu snaží z oběti vylákat peníze tím, že požaduje provedení platby v bitcoinech na uvedenou Bitcoinovou adresu a na oplátku nabízí dvojnásobek odeslané sumy. Tyto peníze však již nikdy nepošle.

Tento nástroj umí tento druh webů detekovat a s pomocí crawleru a sady vyhledávačů tyto weby vyhledává. Nalezené weby jsou následně monitorovány s cílem sledování plateb a použitých Bitcoinových adres. Tento systém se od ostatních odlišuje tím, že nepoužívá analýzu transakcí na blockchainu, nýbrž aktivně vyhledává tento typ podvodných webů. Díky tomu je schopen nalézt Bitcoinové adresy, které mají velmi málo či dokonce žádné přijaté transakce. Více než polovina adres, na které byly nakonec nějaké prostředky odeslány, byly detekovány ještě před tím, než na ně byla odeslána první platba. V období od listopadu 2019 do února 2020 bylo s tímto druhem podvodu nalezeno více než 1300 adres na více než 500 doménách. Celkově bylo na tyto adresy odesláno více než 5 milionu amerických dolarů. Průměrná transakce činila 47,3 dolaru. Data spojená s touto prací je možné nalézt na <https://bit.ly/bgsieeesb2020>. [1]



Obrázek 2.1: Příklad Bitcoinového generátoru [1]

2.1.2 Platforma pro shromažďování kryptoměnových adres

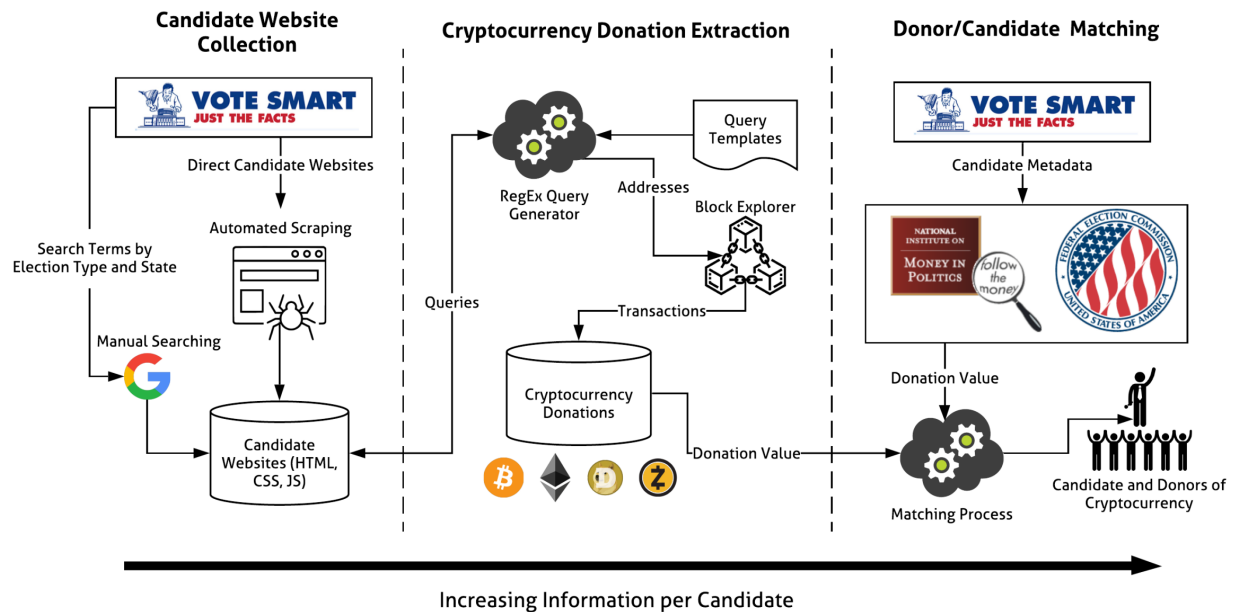
Tato práce se zaměřuje na vývoj platformy pro sběr a zobrazování metadat o kryptoměnových adresách nejen z povrchového, ale také z temného webu. Součástí této platformy je také crawler, který na začátku projde všechny weby, které by měly být prohledány a následně ukládá URL adresy v nich obsažené včetně příslušných metadat. Tyto adresy pak dále prochází a případně z nich získává potřebné informace pomocí scraperů. Tyto scrapery analyzují webové stránky podle procházené adresy a extrahují zajímavé metadata. K scrapování je použita PHP knihovna Goutte. [2]

2.1.3 Údržba úložiště Bitcoinových adres prostřednictvím cíleného procházení webu

Tato diplomová práce používá pravidelné crawlování uživatelem zadaných kryptoměnových stránek, jejichž výsledky ukládá do XML souboru, který je možné exportovat do libovolné databáze. Získané adresy a metadata mohou být použity při vyšetřování trestných činů spojených s Bitcoinem. Data o transakcích získaných adres jsou poté dostupné na Bitcoinovém blockchainu. [3]

2.1.4 Odhalování nezákonných kryptoměnových příspěvků v politických kampaních

Tato práce se zaměřuje na americké politické kandidáty, kteří v rámci svých kampaní přijímají kryptoměny. Analyzuje aktuální stav v kampaních napříč USA a shromažďuje data o webových stránkách kandidátů, kryptoměnových adresách a zprávách o darech zaslaných federálním regulačním orgánům. Při porovnání transakcí nad danými adresami a potvrzeními od federální volební komise je možné pozorovat, že mnoho kandidátů své kryptoměnové dary nehlásí a ti kteří tak činí, tak velmi často chybně. Někteří kandidáti také výrazně podhodnocují přijaté dary. Práce také uvádí doporučení pro vytvoření systému, kde by bylo možné využít kryptoměny pro politické dary, aniž by mohlo dojít k nesprávnému zacházení s finančními prostředky či vyhýbání se federálním směrnicím. V rámci této centrální platformy by dárce zaregistrovali své kryptoměnové adresy, které by poté sloužili k ověření identity dárce.



Obrázek 2.2: Proces shromažďování dat o politických kandidátech USA [4]

K analýze využití kryptoměnových darů byly shromážděny data z webových stránek (získaných ze stránek VoteSmart [5]) 4808 kandidátů, které byly doplněny o ručně vyhledané stránky na Google pomocí parametrů získaných od VoteSmart. Všechny ručně získané stránky pak místo adres používaly službu BitPay. Pomocí automatizačního nástroje Selenium poté bylo úspěšně staženo 4637 stránek. Celkem bylo nalezeno 20 kryptoměnových adres nad kterými proběhlo 157 transakcí. [4]

2.1.5 Detekce Bitcoinových Ponziho schémat pomocí dolování dat

Tato práce se zaměřuje na dolování dat s cílem detekovat Ponziho schémata, která využívají Bitcoin. Ponziho schéma je označení pro podvodnou investici, ve které jsou uživatelům vypláceny prostředky pomocí investic nových uživatelů. Celý podvod zdánlivě funguje až do chvíle, kdy již není možné najít nové investice. Navzdory tomu, že jsou Ponziho schémata v mnoha zemích nezákonná, se dnes po celém světě rozšiřují a využívají k tomu právě Bitcoin. Kybernetičtí zločinci zneužívají Bitcoinové pseudonymity k provádění svých podvodů, jelikož je složité vysledovat vlastníky daných adres.

K získání Bitcoinových adres je provedeno ruční vyhledávání na stránkách Reddit a bitcoin-talk.org, což jsou hlavní diskuzní fóra o Bitcoinu. Práce se zaměřuje na podfórum „Hazardní hry: Hry založené na investorech“ na bitcointalk.org, kde podvodníci obvykle propagují Ponziho schémata jako „investiční programy s vysokým výnosem“ nebo různé hazardní hry. Pouze v několika případech tyto reklamy výslovně obsahují Bitcoinovou adresu. Obvykle je však k získání dané adresy nutné navštívit konkrétní webovou stránku, na které je dané Ponziho schéma provozováno. Mnoho z těchto webových stránek však již není online, a tak je snaha obnovit jejich snímek prostřednictvím archive.org. Na každé získané stránce jsou pak ručně vyhledány jednotlivé adresy. Toto vyhledávání je dále doplněno o adresy vypsane na blockchain.info/tags, což je webová stránka, která umožňuje uživatelům označovat Bitcoinové adresy. Většina označených adres také obsahuje odkaz na stránku na které se objevuje. V rámci dané práce byl také vyvinut crawler, který automaticky prohledává získané stránky a ohodnocuje je na základě počtu obsažených slov spojených s Ponziho schématem. K tomuto účelu je použit slovník obsahující slova jako např. „Ponzi“, „zisk“, „multiplikátor“ či „investice“. Crawler prošel více než 1500 stránek (souvisejících s přibližně 3500 označeními) a zjistil, že přibližně 900 z nich obsahuje nějaké slovo související s tímto podvodem. Přibližně 600 z těchto stránek však již není přístupných a to ani přes archive.org. [6]

2.1.6 Analýza útoku Man-in-the-middle nad Bitcoinovou adresou

Tato práce se zabývá analýzou útoku Man-in-the-middle nad Bitcoinovou adresou. Tento útok spočívá v tom, že útočník nahradí původní Bitcoinovou adresu za svou vlastní. Na první pohled nemusí být zřejmé, že k takovéto změně došlo, jelikož jsou kryptoměnové adresy v podstatě dlouhý řetězec různých znaků a čísel. Běžný uživatel si nemá jak ověřit, že daná adresa skutečně patří osobě, které chce své kryptoměny poslat. Vzhledem k tomu pak také tento útok může být i dlouhodoběji úspěšný, alespoň tedy za předpokladu, že příjemce neočekává relativně pravidelný příjem kryptoměn.

Analýza spočívá v procházení stránek, které obsahují Bitcoinové adresy a tedy přijímají platby v bitcoinech. Dále pak určuje kolik adres je vystaveno útoku Man-in-the-middle. Webové stránky, které byly při analýze procházeny jsou získávány ze stránky blockchain.info, která obsahuje u jednotlivých adres také stránky na kterých se tyto adresy nacházejí. K následnému vyhledání Bitcoinové adresy na dané stránce je použito filtrování pomocí regulárních výrazů. Součástí analýzy je také to, kolik adres používá HTTP/HTTPS během transakcí a kolik adres je aktivních či neaktivních. Celkově bylo nalezeno přibližně 10 tisíc adres (z toho okolo 8 tisíc aktivních), nad kterými proběhlo okolo 750 tisíc transakcí o celkové hodnotě přibližně 850 tisíc bitcoinů. [7]

2.2 Databáze adres a nahlášených podvodů

V rámci přípravy databáze Bitcoinových adres a informací s nimi spojených bylo nejprve potřeba najít již existující zdroje. Ideálně by pak každý zdroj měl obsahovat také nějaké další informace a metadata, které by bylo možné dále využít.

Při průzkumu existujících prací na téma Bitcoinu a dalších kryptoměn bylo nalezeno několik prací, které se zmiňují o nějaké interní databázi adres. Pouze jedna z těchto prací (Bitcoin Generator Scam [1]) však obsahovala odkaz k datasetu, který navíc obsahoval pouze seznam adres bez dalších informací k jednotlivým adresám.

Existující databáze adres samozřejmě nebyly sbírány pouze z akademických zdrojů, ale také pomocí vyhledávačů povrchové části internetu (vyhledávač Google). Tato akce již byla o něco úspěšnější, jelikož se podařilo najít celkem 9 dalších zdrojů. Kvalita zdrojů se ve velkém liší, jak v počtu adres, tak v počtu informací ke konkrétním adresám, až po metody přístupu k těmto datům.

Vzhledem k tomu, že veškerá historie Bitcoinových transakcí je zaznamenána na Bitcoinovém blockchainu, tak jediné adresy, které mohou být součástí transakcí či mohou obsahovat nějaký zůstatek jsou právě adresy, které se někdy objevily na blockchainu.

Zdroj LoyceV zpracovává blockchainové data a poskytuje seznam všech adres v pořadí dle jejich výskytu na blockchainu či seznam všech adres s nenulovým zůstatkem v pořadí dle výše zůstatku. Seznam nenulových adres je identický se seznamem poskytovaným službou BlockChair, jelikož LoyceV tyto data čerpá právě z BlockChair. Nevýhodou tohoto zdroje je, že se o jeho udržování stará jednatel, který může tuto činnost kdykoliv ukončit. V takovém případě by bylo nutné vytvořit vlastní řešení pro zpracování blockchainových dat. Dále také může být problém rozsah dat, jelikož počet všech adres dosahuje více než 1 miliardy a tedy může být obtížné tyto adresy do databáze uložit. [8]

BlockChair kromě již zmíněných nenulových adres obsahuje také data o Bitcoinových blocích, transakcích, vstupech a výstupech. Blockchair je engine pro vyhledávání a analýzu na blockchainu Bitcoinu a dalších kryptoměn. Tato služba si dává za cíl se stát Googlem pro blockchainya. Data poskytuje v souborech, které jsou aktualizovány jednou denně a případně také pomocí API. Hlavním problémem je, že bezplatná API umožňuje max. 1 440 požadavků za den a stahování souborů má limitovanou rychlost na 100kB/s, což vede k tomu, že stahování dat, oproti předchozí zmíněné službě, trvá i více než několik hodin. [9]

BitcoinAbuse je veřejná databáze Bitcoinových adres používaných podvodníky, hackery a zločinci. BitcoinAbuse umožňuje nahlášovat Bitcoinové adresy a tyto nahlášení poté stahovat pomocí API v podobě CSV souboru. Tyto soubory obsahují nahlášení buď za 1 den, 30 dní či za celou historii, nicméně poslední možnost není aktuálně v provozu. Data jde každopádně kromě API také ručně vyhledat přímo na webu. [10]

CheckBitcoinAddress je stránka za kterou stojí firma Apirone, která poskytuje služby pro příjem, zpracování a posílání plateb v kryptoměnách. CheckBitcoinAddress čerpá Bitcoinové adresy ze služby BitcoinAbuse a na webu poskytuje seznam veškerých nahlášení k daným adresám ve kterém je možné ručně vyhledávat. Nevýhodou této služby je absence API. [11]

BitcoinAIS je služba, která sleduje Bitcoinové adresy používané v nevyžádaných emailech, ransomware, malware a k jiným podvodům. Na webu je možné ručně vyhledávat v seznamu posledních několika nahlášení. Nevýhodou této služby je, že neposkytuje API pro vyhledávání a také, že zobrazuje pouze omezený počet nahlášení. [12]

Bitcoin Generator Scam je dataset získaných dat v rámci práce „An Automatic Detection and Analysis of the Bitcoin Generator Scam“. [1] Tato práce vznikla s cílem nalézt podvody typu „Bitcoinového generátoru“, které se objevují na stránkách nabízejících znásobení odeslané částky v bitcoinech. Datasets jsou poskytovány v textovém souboru a obsahují samostatné soubory pro Bitcoinové adresy (adresy s transakcemi a bez transakcí) a také jeden pro adresy jiných kryptoměn. Nevýhodou je, že tyto adresy neobsahují žádné další metadata jako např. čas jejich získání či konkrétní zdroj. Poskytuje pouze seznam všech podvodných domén. [13]

CryptoBlacklist je webová stránka, která obsahuje vyhledávač, který pro zadanou kryptoměnovou adresu vypíše, zda je obsažena v databázi existujících nahlášení či nikoliv. Nevýhodou této služby je, že obsahuje pouze limitovaný počet posledních nahlášených adres a vyhledávat je možné pouze skrze integrovaný vyhledávač. [14]

CryptoScamDB je open-source web provozovaný společností MyCrypto, která se zaměřuje na správu účtů na síti Ethereum. Služba CryptoScamDB poskytuje API, která vrací JSON, ve kterém jsou všechny adresy, které byly nahlášený. U každé adresy jsou pak jednotlivé nahlášení, ve kterých se daná adresa objevuje. Nevýhodou tohoto zdroje je, že obsahuje nejen Bitcoinové adresy, ale i adresy dalších kryptoměn, kterých je většina. [15]

SeeKoin se prezentuje jako vyhledávač pro Bitcoin a další kryptoměny. Na webu obsahuje seznam adres spojených s nahlášenými podvody, ve kterém je možné ručně vyhledávat. Nevýhodou tohoto zdroje je, že obsahuje relativně špatný stránkovací systém, takže je u něj těžké určit, kolik opravdu obsahuje nahlášení. [16]

Cryptscam částečně čerpá z BitcoinAbuse a vypisuje pouze 1 000 posledních nahlášení nejen Bitcoinových adres. Zda byla adresa součástí nahlášení je nicméně možné vyhledat pomocí integrovaného vyhledávače. Nevýhodou tohoto zdroje je omezený počet posledních nahlášení a vyhledávání dostupné pouze skrze integrovaný vyhledávač. [17]

BitcoinWhosWho je zdroj, který poskytuje pouze integrovaný vyhledávač a tedy není možné zjistit kolik adres obsahuje bez pokusu o vyhledání všech adres vyskytujících se na blockchainu. Tento zdroj pak obsahuje několik informací o vyhledané adrese, ale nahlášení k dané adrese příliš informací neobsahují. [18]

2.3 Shrnutí relevantních zdrojů

Statistiky výše uvedených zdrojů jsou shrnuty v tabulce 2.1, která jednotlivé zdroje seřazuje dle potenciálního počtu Bitcoinových adres. Data k jednotlivým zdrojům jsou platná k 1.11.2022.

| Zdroj | Počet adres/nahlášení | Pouze BTC | Přístup |
|-----------------------------|--------------------------------|-----------|-----------------|
| LoyceV [8] | 1 043 489 822 unikátních adres | ANO | Soubor (TXT) |
| BitcoinAbuse [10] | 307 641 nahlášení | NE | API (CSV) / Web |
| CheckBitcoinAddress [11] | 88 868 unikátních adres | NE | Web |
| CryptoBlacklist [14] | 37 881 unikátních adres | NE | Vyhledávač |
| Bitcoin Generator Scam [13] | 6 395 unikátních adres | NE | Soubor (TXT) |
| BitcoinAIS [12] | 5 699 nahlášení | NE | Web |
| CryptoScamDB [15] | 4 478 unikátních adres | NE | API (JSON) |
| Cryptscam [17] | 1 000 nahlášení | NE | Vyhledávač |
| SeeKoin [16] | neurčitý počet adres | ANO | Web |
| BitcoinWhosWho [18] | neurčitý počet adres | ANO | Vyhledávač |

Tabulka 2.1: Zdroje kryptoměnových adres

Jak tedy bylo zjištěno, aktuálně na internetu není mnoho kvalitních zdrojů a dokonce ani práce zabývající se podvody spojenými s Bitcoinem nezveřejňují vlastní databáze adres. Vytvářená databáze by tedy měla obsahovat pokud možno všechny Bitcoinové adresy z blockchainu získané z LoyceV a pak dále veškeré informace z dalších zdrojů získané ať už pomocí jejich API či vlastního crawleru. Databáze by pak měla být finálně zveřejněna, aby mohla sloužit k dalšímu využití a snížení aktuálního nedostatku relevantních zdrojů.

Kapitola 3

Analýza požadavků

3.1 Požadavky

| ID | Popis | Typ | Druh požadavku | Případ užití | Doména |
|----------|---|-----|----------------|--------------|--------|
| N1 | Uživatelské účty z pohledu systému | N | – | – | SW |
| N1–P1 | Systém umožňuje zobrazení seznamu uživatelů pouze administrátorům | P | Funkční | – | SW |
| N1–P2 | Systém zamezuje přístup k API uživatelům, kteří mají naplněn limit počtu použití | P | Funkční | – | SW |
| N1–P2–I1 | Uživatel nemá přístup k API, jelikož v daném časovém rozmezí vyčerpal stanovený limit počtu použití | I | – | – | SW |
| N2 | Uživatelské účty z pohledu administrátora | N | – | – | SW |
| N2–P1 | Systém umožňuje zobrazení seznamu uživatelů | P | Funkční | 12 | SW |
| N2–P2 | Systém umožňuje upravení role uživatele | P | Funkční | 14 | SW |
| N3 | Uživatelské účty z pohledu registrovaného uživatele | N | – | – | SW |
| N3–P1 | Systém poskytuje informace k danému API tokenu | P | Funkční | 16 | SW |
| N4 | Uživatelské účty z pohledu nepřihlášeného uživatele | N | – | – | SW |
| N4–P1 | Systém umožňuje vyhledání adresy | P | Funkční | 1 | SW |
| N4–P2 | Systém zamezuje přístup k API, jelikož uživatel nevlastní API token | P | Funkční | – | SW |

Tabulka 3.1: Požadavky pro uživatelské účty

3.2 Případy užití

Případy užití se liší dle rozhraní, ve kterém je k systému přistupováno. Vzhledem k tomu, že web i API poskytují podobné, ne však identické data, k jejichž získání vyžadují odlišné identifikační údaje, tak lze tyto případy užití identifikovat z pohledu 2 podsystémů a to i přestože v základu pracují s identickou databází.

3.2.1 Případy užití webu

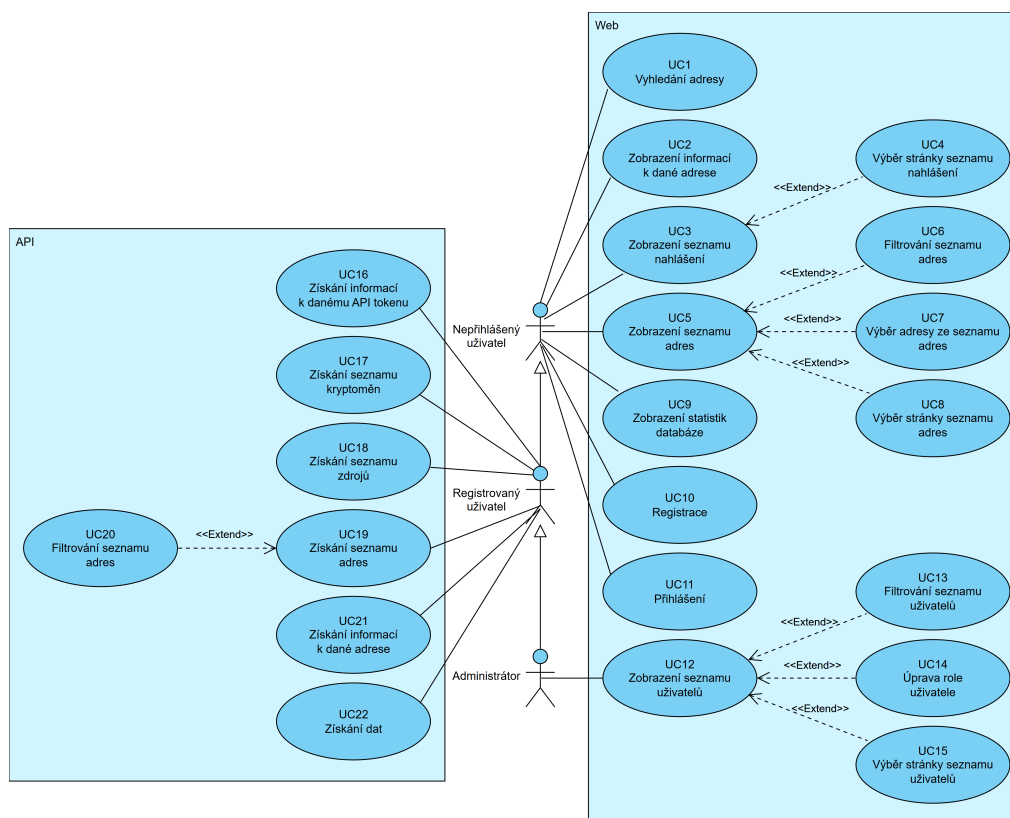
V rámci webu je hlavní rozdíl v tom, zda uživatel vystupuje v roli administrátora či nikoliv. Administrátor má totiž navíc od běžného uživatele přístup k zobrazení seznamu uživatelů a také k úpravě rolí jednotlivých uživatelů. Může tedy změnit roli libovolného uživatele na jednu ze 4 dostupných variant – host, uživatel, insider a administrátor. Tyto uživatelské role pak mimo jiné ovlivňují především to, k jakým datům se ve finále uživatel na webu či pomocí API dostane, nicméně více o omezení přístupu k datům daném těmito rolemi je uvedeno v sekci 5.4.3.

3.2.2 Případy užití API

API poskytuje data pouze registrovaným uživatelům webu a to jednoduše tak, že k přístupu do API je potřeba mít API token, který musí být součástí odesílaného požadavku. Tento token se generuje při registraci a určuje, k jakým datům má uživatel přístup a to na stejném principu, který již byl zmíněn v sekci 3.2.1 a je dále podrobněji rozvinut v sekci 5.4.3. Kromě toho pak tento token společně s rolí uživatele ovlivňují to, jak často a kolik celkově za určitý čas může být s daným tokenem odesláno požadavků.

3.2.3 Shrnutí případů užití

Následující diagram 3.1 vychází z již zmíněných největších rozdílů mezi uživateli. Nepřihlášený uživatel nemá přístup k API ani k zobrazení či úpravě uživatelů. Registrovaný uživatel k API přístup má díky API tokenu, který se při registraci generuje. Administrátor má pak přístup ke všem případům užití.



Obrázek 3.1: Diagram případů užití

3.2.4 Vybraný scénář případu užití

Scénář UC14: Úprava role uživatele

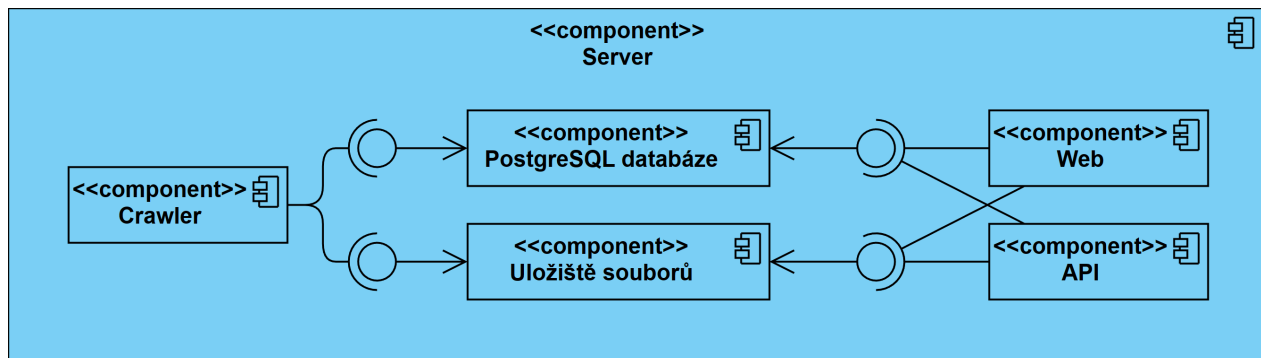
1. Administrátor se přihlásí do systému.
2. Systém ověří, že daný uživatelský účet v systému existuje.
3. Systém administrátora přesměruje na stránku uživatele.
4. Administrátor klikne na odkaz s textem „Accounts“
5. Systém ověří, že v systému existuje alespoň jeden uživatel.
6. Systém vypíše seznam uživatelů.
7. Administrátor u vybraného uživatele změní roli.
8. Administrátor potvrdí změnu stisknutím tlačítka „Edit“.
9. Systém ověří, že zadané údaje jsou v pořádku.
10. Systém změní roli daného uživatele.

Rozšíření scénáře UC14: Úprava role uživatele

- 2.a) Systém zjistí, že daný uživatelský účet v systému neexistuje.
 - 2.a.1) Systém zobrazí chybovou hlášku: „Account does not exist.“.
 - 2.a.2) Příklad užití pokračuje krokem 1.
- 2.b) Systém zjistí, že pro daný email bylo zadáno nesprávné heslo.
 - 2.b.1) Systém zobrazí chybovou hlášku: „Account does not exist.“.
 - 2.b.2) Příklad užití pokračuje krokem 1.
- 5.a) Systém zjistí, že v systému není ani jeden uživatel.
 - 5.a.1) Systém vypíše hlášku: „No account has been found.“.
 - 5.a.2) Příklad užití končí.
- 9.a) Systém zjistí, že daný uživatel v systému neexistuje.
 - 9.a.1) Systém zobrazí chybovou hlášku: „Account does not exist.“.
 - 9.a.2) Příklad užití pokračuje krokem 5.
- 9.b) Systém zjistí, že daný uživatel je již posledním administrátorem v systému.
 - 9.b.1) Příklad užití pokračuje krokem 5.

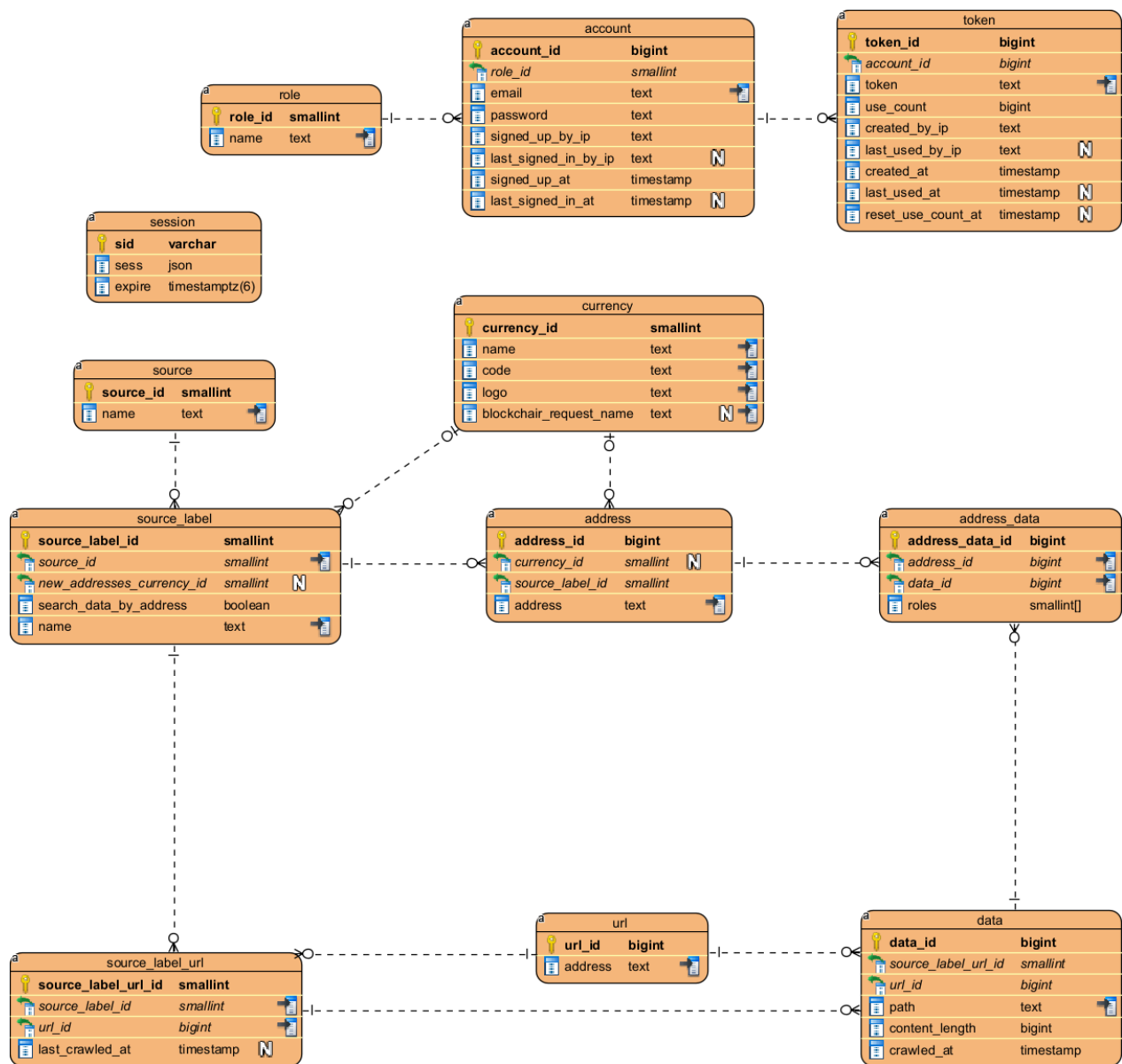
3.3 Architektura

Systém je postaven na architektuře Klient-Server. Data se na serveru zpracovávají, organizují a uchovávají. Server pak přístup k těmto datům poskytuje pomocí webu a API. Klientem může být např. webový prohlížeč, který po odeslání požadavku na server získá odpovídající výstup v podobě HTML stránky.



Obrázek 3.2: Diagram komponent systému

3.4 Návrh databáze



Obrázek 3.3: Diagram vztahu entit databáze

3.4.1 Datový slovník

Zatímco diagram vztahu entit databáze graficky a přehledně znázorňuje především propojení a velikost databázové struktury, tak datový slovník slouží pro podrobnější popis jednotlivých tabulek a atributů a jejich využití v systému.

3.4.1.1 Tabulka account

Tabulka account uchovává informace o uživatelských účtech na webu. Přihlašování umožňují atributy email a password, které jsou porovnávány jak při pokusu o přihlášení, tak také při pokusu o registraci. Následně jsou zde informační atributy zaznamenávající IP uživatele a čas registrace i posledního přihlášení, což jsou údaje, které se mohou hodit k nějaké další kontrole či statistice. Velký význam pak má identifikátor role, který určuje k jakým údajům má uživatel přístup ať už na webu či pomocí API.

| Atribut | Dat. typ | Klíč | Null | Význam |
|----------------------|-----------|------|------|------------------------------------|
| account_id | BigSerial | PK | Ne | Identifikátor uživatele |
| role_id | SmallInt | FK | Ne | Identifikátor role |
| email | Text | UQ | Ne | Email uživatele |
| password | Text | | Ne | Heslo uživatele (hash) |
| signed_up_by_ip | Text | | Ne | IP uživatele (registrace) |
| last_signed_in_by_ip | Text | | Ano | IP uživatele (poslední přihlášení) |
| signed_up_at | Timestamp | | Ne | Čas registrace |
| last_signed_in_at | Timestamp | | Ano | Čas posledního přihlášení |

Tabulka 3.2: Tabulka account

3.4.1.2 Tabulka address

Tabulka address slouží k uchování kryptoměnové adresy. Tato adresa je pak doplněna o identifikátor kryptoměny, který indikuje, zda se jedná o adresu na Bitcoinovém, Ethereumovém či jiném blockchainu. Tento údaj pak umožňuje podle těchto kryptoměn adresy na webu filtrovat. Identifikátor podkategorie zdroje říká, kde byla daná adresa při crawlování poprvé nalezena, což je údaj, který systém aktuálně nevyužívá, ale mohl by se potenciálně hodit při jeho rozšiřování.

| Atribut | Dat. typ | Klíč | Null | Význam |
|-----------------|-----------|------|------|-----------------------------------|
| address_id | BigSerial | PK | Ne | Identifikátor adresy |
| currency_id | SmallInt | FK | Ano | Identifikátor kryptoměny |
| source_label_id | SmallInt | FK | Ne | Identifikátor podkategorie zdroje |
| address | Text | UQ | Ne | Kryptoměnová adresa |

Tabulka 3.3: Tabulka address

3.4.1.3 Tabulka address_data

Tabulka address_data propojuje kryptoměnovou adresu s daty získanými crawlováním a to pomocí identifikátorů adresy a dat. Atribut roles obsahuje všechny identifikátory rolí, které mají k daným datům přístup.

| Atribut | Dat. typ | Klíč | Null | Význam |
|-----------------|------------|---------------------|------|--------------------------------------|
| address_data_id | BigSerial | PK | Ne | Identifikátor propojení adresy a dat |
| address_id | BigInt | FK, UQ ₁ | Ne | Identifikátor adresy |
| data_id | BigInt | FK, UQ ₁ | Ne | Identifikátor dat |
| roles | SmallInt[] | | Ne | Pole identifikátorů rolí |

Tabulka 3.4: Tabulka address_data

3.4.1.4 Tabulka currency

Tabulka currency obsahuje informace o kryptoměnách se kterými systém pracuje. Seznam těchto kryptoměn systém přejímá ze seznamu podporovaných blockchainů na stránce blockchair.com. V této tabulce jsou pak zaznamenány název a kód kryptoměny, dále cesta k souboru s jejím logem a URL řetězec z již zmíněné stránky. Tento řetězec slouží k rozpoznání kryptoměny u adres, které jsou obsaženy na zdrojích, které tuto informaci neposkytují. Jedná se však pouze o altcoinové adresy, jelikož všechny adresy na Bitcoinovém blockchainu jsou v systému již obsaženy crawlováním jednoho ze zdrojů.

| Atribut | Dat. typ | Klíč | Null | Význam |
|-------------------------|-------------|------|------|--------------------------------------|
| currency_id | SmallSerial | PK | Ne | Identifikátor kryptoměny |
| name | Text | UQ | Ne | Název kryptoměny |
| code | Text | UQ | Ne | Kód kryptoměny |
| logo | Text | UQ | Ne | Cesta k souboru s logem kryptoměny |
| blockchair_request_name | Text | UQ | Ano | URL řetězec kryptoměny na Blockchair |

Tabulka 3.5: Tabulka currency

3.4.1.5 Tabulka data

Tabulka data slouží k uchování metadat souboru získaného při crawlování. Obsahuje identifikátor URL podkategorie zdroje, která říká, z jakého zdroje byl daný soubor získán a také identifikátor URL, který obsahuje konkrétní URL, ze které byl soubor stažen. Cesta k souboru s daty pak umožňuje jeho načtení a zpracování na webu. Délka obsahu souboru a čas crawlování slouží pouze jako užitečná informace navíc.

| Atribut | Dat. typ | Klíč | Null | Význam |
|---------------------|-----------|------|------|---------------------------------------|
| data_id | BigSerial | PK | Ne | Identifikátor dat |
| source_label_url_id | SmallInt | FK | Ne | Identifikátor URL podkategorie zdroje |
| url_id | BigInt | FK | Ne | Identifikátor URL |
| path | Text | UQ | Ne | Cesta k souboru s daty |
| content_length | BigInt | | Ne | Délka obsahu souboru |
| crawled_at | Timestamp | | Ne | Čas crawlování |

Tabulka 3.6: Tabulka data

3.4.1.6 Tabulka role

Tabulka role zaznamenává uživatelské role, které mají význam jak z pohledu oprávnění na webu, tak z pohledu přístupu k datům.

| Atribut | Dat. typ | Klíč | Null | Význam |
|---------|-------------|------|------|--------------------|
| role_id | SmallSerial | PK | Ne | Identifikátor role |
| name | Text | UQ | Ne | Název role |

Tabulka 3.7: Tabulka role

3.4.1.7 Tabulka session

Tabulka session slouží k uchování sezení uživatele webu a je používána Node.js balíčkem express-session.

| Atribut | Dat. typ | Klíč | Null | Význam |
|---------|----------------|------|------|----------------------|
| sid | Varchar | PK | Ne | Identifikátor sezení |
| sess | JSON | | Ne | Data sezení |
| expire | Timestamptz(6) | | Ne | Čas vypršení sezení |

Tabulka 3.8: Tabulka session

3.4.1.8 Tabulka source

Tabulka source obsahuje názvy zdrojů používaných při crawlování a slouží také pro filtrování adres na webu.

| Atribut | Dat. typ | Klíč | Null | Význam |
|-----------|-------------|------|------|----------------------|
| source_id | SmallSerial | PK | Ne | Identifikátor zdroje |
| name | Text | UQ | Ne | Název zdroje |

Tabulka 3.9: Tabulka source

3.4.1.9 Tabulka source_label

Tabulka source_label zaznamenává podkategorie zdrojů a to z toho důvodu, že každý zdroj může obsahovat více typů dat, které nemusí být součástí stejné skupiny i přestože jsou získány ze stejného zdroje. Jako příklad lze uvést Bitcoin Generator Scam, který obsahuje zdroj pouze Bitcoinových adres a pak také zdroj adres i jiných kryptoměn. Pokud se pro danou podkategorii na daném zdroji nachází pouze adresy jedné kryptoměny, pak je tato kryptoměna uvedena v příslušném identifikátoru, který je použit v tabulce address pro označení adres přidáných tímto zdrojem. V případě, že se má při crawlování použít webový vyhledávač, tak je tato informace zaznamenána v atributu search_data_by_address.

| Atribut | Dat. typ | Klíč | Null | Význam |
|---------------------------|-------------|---------------------|------|---|
| source_label_id | SmallSerial | PK | Ne | Identifikátor podkategorie zdroje |
| source_id | SmallInt | FK, UQ ₁ | Ne | Identifikátor zdroje |
| new_addresses_currency_id | SmallInt | FK | Ano | Identifikátor kryptoměny (pro nově přidané adresy) |
| search_data_by_address | Boolean | | Ne | Použít vyhledávač (crawlování) |
| name | Text | UQ ₁ | Ne | Název podkategorie zdroje |

Tabulka 3.10: Tabulka source_label

3.4.1.10 Tabulka source_label_url

Tabulka source_label_url obsahuje URL podkategorií zdrojů, jelikož i podkategorie zdrojů mohou mít více URL. Příkladem je zdroj Bitcoin Generator Scam, který obsahuje dvě URL se seznamy Bitcoinových adres. Toto oddělení není pro tuto práci důležité, takže tyto URL patří do stejné podkategorie. Jakmile je crawlování pro danou URL podkategorie dokončeno, tak je zaznamenán čas posledního crawlování.

| Atribut | Dat. typ | Klíč | Null | Význam |
|---------------------|-------------|---------------------|------|---------------------------------------|
| source_label_url_id | SmallSerial | PK | Ne | Identifikátor URL podkategorie zdroje |
| source_label_id | SmallInt | FK, UQ ₁ | Ne | Identifikátor podkategorie zdroje |
| url_id | BigInt | FK, UQ ₁ | Ne | Identifikátor URL |
| last_crawled_at | Timestamp | | Ano | Čas posledního crawlování |

Tabulka 3.11: Tabulka source_label_url

3.4.1.11 Tabulka token

Tabulka token uchovává informace o API tokenu. Kromě samotného tokenu zaznamenává uživatele, se kterým je spojena, počet použití tokenu, IP uživatele a čas při vytvoření a posledním použití a také čas, kdy se počet použití resetuje. Počet použití a čas resetování zajišťují to, že na základě uživatelské role je daná API limitovaná tak, aby nemohla být využívána přes limit na úkor stability systému.

| Atribut | Dat. typ | Klíč | Null | Význam |
|--------------------|-----------|------|------|---------------------------------|
| token_id | BigSerial | PK | Ne | Identifikátor API tokenu |
| account_id | BigInt | FK | Ne | Identifikátor uživatele |
| token | Text | UQ | Ne | API token |
| use_count | BigInt | | Ne | Počet použití |
| created_by_ip | Text | | Ne | IP uživatele (vytvoření) |
| last_used_by_ip | Text | | Ano | IP uživatele (poslední použití) |
| created_at | Timestamp | | Ne | Čas vytvoření |
| last_used_at | Timestamp | | Ano | Čas posledního použití |
| reset_use_count_at | Timestamp | | Ano | Čas resetování počtu použití |

Tabulka 3.12: Tabulka token

3.4.1.12 Tabulka url

Tabulka url obsahuje unikátní URL používané při crawlování.

| Atribut | Dat. typ | Klíč | Null | Význam |
|---------|-----------|------|------|-------------------|
| url_id | BigSerial | PK | Ne | Identifikátor URL |
| address | Text | UQ | Ne | URL adresa |

Tabulka 3.13: Tabulka url

Kapitola 4

Implementace crawleru

Prvním implementovaným nástrojem je crawler. Tento nástroj je navržen v programovacím jazyce Python a má za cíl nejprve vytvořit databázi, přidat do ní její uživatele a výchozí data. Dále nastavit určité parametry pro optimalizaci následného vkládání dat a tyto parametry potvrdit restartováním procesu databázového serveru, k čemuž je potřeba, aby nástroj běžel v režimu správce.

Hlavním cílem, který následuje, je pak procházení předem daných zdrojů dat, ze kterých lze s pomocí vhodného nastavení hlavičky a frekvence požadavků získat nahlášení spojená s Bitcoinovými, ale i jinými kryptoměnovými adresami. Vzhledem k tomu, že je potřeba zajistit dlouhodobě funkční běh tohoto nástroje, tak je kromě nastavení hlavičky a frekvence požadavků vhodné splňovat pravidla crawlování, které si daný zdroj vymezuje v souboru `robots.txt`. Aby pak data byly co nejdříve uloženy do databáze nejen pro zajištění jejich zachování, ale také pro jejich následné použití na webu, tak je proces vkládání optimalizován na úrovni SQL příkazů, které doplňují vlákna, které umožňují souběžné crawlování více zdrojů současně.

Samotné ukládání dat je pak dále rozepsáno podrobněji a netýká se pouze databáze, která obsahuje důležitá metadata, ale také disku, jelikož na disk se ukládají samotné raw data z webů a API. Tyto data mají obvykle podobu souborů ve formátu HTML, TXT či JSON.

Protože cílem celého řešení má být vyhledání nalezených dat ke konkrétním kryptoměnovým adresám, tak je potřeba tyto adresy s danými daty propojit. Nezanedbatelná informace je poté také určení k jaké kryptoměně se daná adresa vztahuje, což je u Bitcoinu jednodušší díky tomu, že dané řešení pracuje se všemi existujícími Bitcoinovými adresami. Aby mohl nástroj běžet dlouhodobě a samostatně bez větších zásahů, tak je potřeba zajistit jeho automatizovaný běh, čemuž se věnuje poslední sekce této kapitoly.

4.1 Správa databáze

4.1.1 Vytvoření / resetování databáze

Jednou z prvních věcí, kterou crawler provádí po jeho spuštění, je příprava databáze. Jelikož je v rámci životního cyklu řešení crawler prvním nástrojem, který s databází pracuje, pak je příhodné do něj přidat její zavedení či resetování, aby tento proces nemusel být zajišťován manuálně. Minimálně při vývoji je totiž obnovování a opětovné nastavování databáze velmi časté a vzhledem k tomu, že je tento proces pokaždé stejný a také časově náročný, tak je vhodné jej přenechat právě na crawleru, kterému se akorát nastaví požadované parametry. Tato volba velmi urychlila počáteční vývoj a i nyní lze díky ní rychle provádět změny.

4.1.1.1 Vytvoření databáze

Databáze se vytváří pomocí souboru `setup.py`, který načítá nastavení ze souboru `setup.json`. V tomto souboru je možné zadat přístupové údaje k databázi, seznam uživatelů s přístupem k této databázi a také seznam všech tabulek, které do ní mají být přidány. U těchto tabulek je pak možné specifikovat konkrétní parametry sloupců a také zadat výchozí řádky jako např. předem známý seznam podporovaných kryptoměn. Během vytváření databáze se pak aplikují také určité optimalizační parametry, které jsou více rozvedeny v sekci 4.1.4.

Vzhledem k tomu, že s některými parametry tohoto souboru je počítáno např. při crawlování, tak je potřeba úpravy provádět opatrně, jelikož i změna pořadí či názvu některých nastavení by mohla ovlivnit stabilitu daného řešení a vyžadovala by změnu i na straně zdrojového kódu.

4.1.1.2 Resetování databáze

Resetování databáze uvádí databázi do původního stavu. Tento proces odebírá veškeré data, tabulky a dokonce i uživatele databáze, aby bylo možné otestovat její opětovné zavedení. Součástí resetování je i odebrání dříve crawlovaných souborů a resetování optimalizačních parametrů.

4.1.2 Uživatelé databáze

Uživatelé databáze se skládají z uživatelského jména a hesla uživatelů, kteří mají přístup k databázovému serveru. Aby byla zajištěna vyšší bezpečnost databáze, tak je snaha se vyhnout používání výchozího účtu vytvořeného při instalaci PostgreSQL, jelikož je u něj vyšší šance na jeho případné neoprávněné použití. Tento účet je tedy při vytváření databáze deaktivován a místo něj jsou vytvořeny účty definované v souboru s nastaveními. Základně jsou zde 2 účty – jeden pro crawler a jeden pro web.

Účet pro crawler má nejvyšší pravomoci, jelikož je potřebuje při nahrávání dat a jelikož běží lokálně na serveru bez externího přístupu, tak je o něco méně náchylnější na zneužití než účet pro web. Web používá účet, který má převážně přístup pouze ke čtení dat, jelikož s menšími výjimkami do databáze nemá důvod zapisovat. Toto zajišťuje, že i kdyby byl daný účet kompromitován, tak by nemělo dojít k ovlivnění crawlovaných dat.

V případě rozšíření daného řešení o další nástroje je pak vhodné přidat další uživatelské účty a to podobným způsobem, jakým byly přidány ty aktuální.

4.1.3 Výchozí data

Při vytváření databáze je často potřeba do ní také nahrát nějaká data. Crawler např. vyžaduje seznam zdrojů, jejichž identifikátory poté používá při propojování s crawlovanými daty a je díky tomu možné data na webu filtrovat dle zdroje. Stejným způsobem pak funguje také seznam podporovaných kryptoměn, na které jsou zase napojeny jednotlivé kryptoměnové adresy. Nechybí pak ani seznam předem známých uživatelských rolí, které ovlivňují nejen přístup k datům, ale také limity API a další. Tyto data lze samozřejmě do databáze přidat manuálně, nicméně stejně jako u vytváření tabulek a uživatelů lze tento proces zautomatizovat a zjednodušit, a proto je také součástí nastavení v souboru `setup.json`.

4.1.4 Optimalizační parametry

Vzhledem k tomu s jak velkým počtem dat crawler pracuje, tak je potřeba využít veškerých možností optimalizace s cílem snížit čas potřebný pro nahrání těchto dat, aby bylo zajištěno jejich zachování. PostgreSQL server umožňuje nastavit několik parametrů, které ovlivňují jeho výkon.

- `synchronous_commit=off` – Asynchronní commit zajišťuje zvýšení výkonu bez rizika poškození dat, které je možné např. u `fsync=off`. Jedná se tedy o kompromis mezi výkonem a bezpečností dat.
- `bgwriter_lru_maxpages=0` – Maximální počet bufferů, které proces zapíše během každé iterační. Zakázáním zapisování se sníží vstupně-výstupní zátěž.
- `log_checkpoints=off` – Zakazuje zaznamenávání logů o záchytných bodech.
- `min_wal_size=4096` – Definuje minimální velikost WAL (transakční logování) na 4 GB.
- `max_wal_size=16384` – Definuje maximální velikost WAL na 16 GB.
- `wal_level=minimal` – Nastaví minimální úroveň informací zaznamenávaných v transakčním logu, což výrazně zrychluje vykonávání databázových operací.
- `max_wal_senders=0` – Zakazuje replikační streamování.

Následující parametry obsahují hodnoty vypočítané pro RAM o velikosti 16 GB. Crawler je schopen velikost RAM získat a tyto hodnoty procentuálně vynásobit, takže není třeba je zadávat manuálně.

- `work_mem=2097151` – Nastavuje pracovní paměť na 4 GB.
- `maintenance_work_mem=1048576` – Nastavuje pracovní paměť pro údržbu na 1 GB.
- `shared_buffers=524288` – Nastavuje paměť sdíleného bufferu na 4 GB.
- `temp_buffers=131072` – Nastavuje paměť dočasného bufferu na 1 GB.

4.1.5 Restartování databázového serveru

Restartováním databázového serveru se potvrdí změny provedené aplikování optimalizačních parametrů zmíněných výše. Ve Windows PostgreSQL server vystupuje v podobě procesu, který může restartovat pouze správce, což je také důvod, proč je nutné při spuštění nástroje potvrdit spuštění v režimu správce. To, zda se má server restartovat je možné změnit vstupním parametrem programu, nicméně bez této změny se ve výchozím stavu databázový server restartuje po vytvoření či resetování databáze. Pro restartování se v Pythonu používá knihovna `ctypes`, která obsahuje přístup k Windows API a jejímu rozhraní Shell (`shell32`).

4.2 Spuštění jako správce

Jak již bylo zmíněno, aby bylo možné restartovat proces databázového serveru, tak je potřeba být v režimu správce. Toho lze dosáhnout tím, že je crawler spouštěný z příkazového řádku, který již je v režimu správce. Pokud příkazový řádek není v režimu správce a je přes něj spuštěný crawler, pak se objeví řízení uživatelských účtů (UAC) a zde je nutné zvolit možnost „Ano“, což aktuální instanci aplikace ukončí a pomocí již zmíněného rozhraní Shell se vytvoří nová instance v režimu správce. V případě, že tato možnost zvolena není, pak je aplikace ukončena.

Kontrola režimu správce byla při spuštění zvolena z toho důvodu, aby nebylo nutné do aplikace zasahovat v její pozdější fázi, jelikož toto potvrzení stačí pouze jednou.

4.3 Procházení zdrojů

Prvním krokem při procházení je načtení seznamu URL ze všech zdrojů, které byly specifikovány v souboru `setup.json` a byly do databáze nahrány při vytváření databáze. Tyto URL slouží jako počáteční bod pro danou crawlovací větev a obvykle nejsou jediné v dané větvi, jelikož obsahem této URL jsou obvykle seznamy kryptoměnových adres či podvodů s nimi spojených a tyto seznamy také obsahují URL, které je potřeba navštívit.

4.3.1 Získání seznamu Bitcoinových adres

Jelikož je hlavní zaměření v této práci především na Bitcoin, tak je na místě u daných adres kontrolovat, zda se jedná o platnou Bitcoinovou adresu či nikoliv. Dalo by se říct, že Bitcoinovou adresu lze poznat jednoduše, jelikož začíná buď „1“, „3“ či „bc1“, což je pravda, nicméně to neznamená, že libovolná sekvence znaků začínajících těmito znaky by kdy mohla na Bitcoinovém blockchainu existovat. Dokonce z toho nelze poznat ani to, zda na blockchainu již někdy byla obsažena, což je pro dané řešení důležité, jelikož identifikaci či podvody spojenými s adresou, která ještě neexistuje není něco čím by se mělo v této práci smysl zabývat. Je zřejmé, že ověření Bitcoinové adresy není programově možné a je třeba využít přímo Bitcoinového blockchainu.

Na internetu existuje několik služeb, které zobrazují informace k Bitcoinové adrese, nicméně aby tyto informace mohl program použít, musel by po jednom tyto adresy kontrolovat odesíláním požadavků, což je nejen neefektivní, ale také časově náročné vzhledem k velkému počtu adres se kterým program pracuje. Jako možné řešení byla zvolena služba LoyceV, která byla již zmíněna v sekci 2.2. Tato služba umožňuje stahovat seznam Bitcoinových adres z blockchainu a tyto data poskytuje na denní a týdenní bázi. Uložením seznamu Bitcoinových adres z blockchainu do databáze pak lze platnost dané Bitcoinové adresy ověřit pouhým porovnáním toho, zda se v databázi nachází, což je řádově rychlejší operace než odesílání požadavku.

Jelikož má soubor se seznamem Bitcoinových adres minimálně několik desítek gigabytů, tak je jeho stažení a uložení do databáze relativně zdlouhavý proces (při testování obvykle trvá i několik hodin), tak je potřeba provést určité optimalizace, které v danou chvíli neumožňují ukládání žádných dalších dat, a proto je prvním a v danou chvíli jediným crawlovaným zdrojem pouze LoyceV.

Jelikož je soubor se seznamem adres příliš velký na to, aby mohl být celý nahrán do paměti, tak je při stahování zpracováván postupně po částech. Denní seznamy adres jsou ve formátu TXT, takže u nich je třeba akorát získat jednotlivé řádky, které obsahují samotné adresy. U týdenního seznamu to však neplatí, jelikož tento seznam je v souboru, která má komprimovaný formát GZ. U stahování tohoto souboru je tedy zpracováván po částech, dekomprimovaný pomocí Python knihovny `isal_zlib` a až následně jsou z textu získávány jednotlivé řádky adres. Oba formáty souborů jsou při stahování rovněž ukládány na disk, jako případná záloha např. kdyby daný zdroj přestal existovat.

4.3.2 Procházení zdrojů se seznamy nahlášení

Zdroje se seznamy nahlášení jsou zdroje, které poskytují data na webu či v souboru bez nutnosti použití vyhledávače. V těchto zdrojích není potřeba vyhledávat postupně adresu po adrese pomocí již získaného seznamu adres, takže je jejich crawling výrazně rychlejší. U těchto zdrojů lze snadněji získat přehled o celkovém počtu adres či nahlášení, jelikož mají obvykle očíslované stránky, kde na každé stránce je stejný počet adres / nahlášení a mnohdy také obsahují tlačítko, kde lze vyčíst číslo poslední stránky. Některé weby dokonce zobrazují vlastní statistiku.

4.3.3 Procházení zdrojů s vyhledávačem

Pro zdroje, které neobsahují seznam adres či nahlášení, či je takový seznam pouze velmi omezený, je nutné použít vyhledávač, který daný zdroj poskytuje. Aby bylo možné tyto zdroje využít, je potřeba mít připravený seznam adres, které se budou vyhledávat, což je také důvod, proč jsou tyto zdroje procházeny jako poslední. Procházení pomocí vyhledávače je pomalé a na prostředky náročné řešení, jelikož se pro každou adresu vytváří samostatný požadavek. Zároveň je také nutné projít všechny adresy z databáze, jelikož není předem jasné, o kterých adresách zdroj obsahuje nějaké informace. Nicméně i přes všechny zmíněné nevýhody není vhodné se těmito zdroji vyhýbat, jelikož i tak je celkový počet zdrojů relativně nízký a každý takový zdroj může obsahovat důležité informace.

4.3.4 Nastavení hlavičky a frekvence požadavků

4.3.4.1 Nastavení hlavičky požadavků

Správné nastavení hlavičky požadavků umožňuje nástroji splynout s požadavky běžných uživatelů. Čím běžnější je v danou chvíli hlavička daného požadavku, tím menší je riziko na případné selektivní zablokování konkrétní hlavičky. Proto by také měla být hlavička aktualizována minimálně vždy při vydání nové verze nejpoužívanějšího prohlížeče, což je v době psaní prohlížeč Google Chrome ve verzi 111.0.0 s hlavičkou obsahující user-agent: „Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/111.0.0.0 Safari/537.36“. Na základě údajů, které poskytuje web useragents.me se aktuálně (17. 3. 2023) jedná o nejběžnější user-agent mezi uživateli osobních počítačů s podílem okolo 38 %.

4.3.4.2 Nastavení frekvence požadavků

Během odesílání požadavků se při vývoji vyskytly určité komplikace, kdy externí server odmítal odpovídat či byl dočasně zahlcen. Toto je při procházení běžný jev způsobený tím, že jsou na server odesílány požadavky ve větší míře, než jakou je server schopen zpracovat. Obecně neexistuje žádné pravidlo, jak dlouho by se mělo mezi jednotlivými požadavky čekat, jelikož to záleží vždy na konkrétním serveru a také velikostí a formátem přenášených dat. Lze však postupným testováním zjistit, při jaké frekvenci nedochází k již zmíněným problémům a taková frekvence pak může fungovat i dlouhodoběji. Samozřejmě se můžou vyskytnout menší výkyvy, kdy bude mít server vyšší návštěvnost i od jiných uživatelů, takže je dobré sledovat, zda se v programu neobjevují chybové hlášky, které by mohly toto naznačovat. Rychlost procházení je samozřejmě možné kdykoliv změnit v souboru `config.json`.

Při vývoji byla jako adekvátní zvolena rychlost přibližně 5 požadavků za sekundu a délka čekání na odpověď každého požadavku na 30 vteřin. Tyto hodnoty nevedly k žádným problémům ani několik hodin od spuštění programu.

4.3.5 Respektování robots.txt

Každý server má právo si určit, ke kterým cestám umožní a ke kterým zakáže přístup a to pomocí souboru `robots.txt` v kořenovém adresáři webu. Server si také může zvolit, jakých konkrétních crawlerů se bude dané pravidlo týkat. Tyto pravidla by pak měla být crawlery dodržována, aby server nezatěžovaly zbytečnými požadavky, které by mohly vést nejen k nestabilitě serveru, ale také k možné blokaci daného crawleru.

Pro tyto účely je v programu využíváno Python knihovny `urllib` a konkrétně její třídy `RobotFileParser`. Tato třída umožňuje zpracovat soubor `robots.txt` a nad jejím objektem lze poté zavolat metodu `can_fetch`, která je pro daný user-agent a URL, kterou chceme procházet, schopna určit, zda k ní máme či nemáme přístup. Pokud soubor `robots.txt` na webu chybí či je jeho obsah prázdný, tak je daný web procházen bez omezení.

4.3.6 Optimalizace pomocí vláken

Vlákna umožňují současně vykonávat více operací na jednou. Jelikož při procházení lze k určitému zdroji přistupovat pouze omezenou rychlostí, aby tento zdroj nebyl příliš zahlcen, tak je možné využít čas a prostředky na to, aby se současně komunikovalo s více zdroji na jednou. Právě z toho důvodu jsou využívána vlákna, jelikož umožňují současné odesílání požadavků na více zdrojů, bez nutnosti čekání na to, než některý z nich odpoví. Nevýhodou může být vyšší zátěž na zařízení, jelikož jsou jeho prostředky využívány mnohem intenzivněji, na druhou stranu však díky efektivnímu využití času trvá průchod všemi zdroji výrazně kratší dobu.

4.3.7 Optimalizace ukládání do databáze

Časově náročné je nejen přenášení dat ze zdroje, ale také jejich ukládání do databáze. Během procházení je do databáze postupně ukládáno velké množství dat a tento proces tak trvá velmi dlouho. Například uložení týdenní aktualizace seznamu Bitcoinových adres tak bez jakýchkoliv optimalizací může trvat i více než týden, jelikož průměrný počet vkládaných záznamů za sekundu se na testovaném zařízení pohybovalo přibližně okolo 100. Použití takové aktualizace pak nemá velký význam, jelikož její data jsou v době dokončení již zastaralá.

První změnou, která výrazně navyšuje počet záznamů za sekundu je odebrání primárních, cizích a unikátních klíčů, kontrol hodnot a toho, zda jsou hodnoty nenulové. Tyto vnitřní kontroly pak nejsou potřeba, jelikož je z externího zdroje zajištěno, že jsou adresy v seznamu unikátní a další hodnoty je možné přidat i bez těchto kontrol. Navíc je veškeré přidávání BTC adres prováděno v jedné transakci, takže jsou tyto kontroly po dokončení ukládání opět obnoveny a po dokončení transakce pak není z tohoto pohledu patrná žádná změna. Touto optimalizací se podařilo zvýšit počet vkládaných záznamů za sekundu přibližně na dvojnásobek.

Dalším způsobem, který byl již částečně zmíněn je zakázání automatických transakcí. Místo nich pak lze vytvářet transakce, které budou dokončeny vždy až po uložení většího množství dat.

Spouštění a ukončování transakce trvá určitý čas, a proto je použití jedné transakce pro jeden vložený řádek velmi neefektivní. Tato optimalizace je nejzásadnější, jelikož díky ní vzrostl počet záznamů za sekundu na více než 200 000.

Poslední změna, která zvýšila u seznamu BTC adres počet záznamů za sekundu až na konečných 250 až 300 tisíc, je změna samotného SQL příkazu. Obvykle se pro ukládání do databáze používá příkaz `INSERT`. V PostgreSQL však existuje ještě příkaz `COPY`, který slouží pro vkládání velkého množství dat. Tento příkaz má jednu nevýhodu, kterou je to, že při libovolné chybě, která se při vkládání objeví, není možné vkládání dokončit. Žádné chyby se však díky kvalitě zdroje LoyceV neočekávají a tak je možné tento příkaz použít.

Kromě ukládání seznamu BTC adres se ukládání i jiná data, která již vyžadují určité kontroly, aby např. neobsahovaly duplikátní informace. Z výše zmíněných optimalizací se tedy používá pouze optimalizace transakcí. Není však jediná. I příkaz `INSERT` je možné zefektivnit a to přidáním až 1 000 řádků v rámci jednoho příkazu `VALUES`. Tento limit není vybrán náhodně. Jedná se o maximální počet řádků, který je v PostgreSQL příkaz `VALUES` schopen pojmout.

4.3.8 Ukládání souborů

Při procházení se stahují a ukládají soubory ve formátu HTML, TXT, JSON či GZ. Tyto soubory jsou používány jednak na webu, ale také v API a to při vyhledání konkrétní kryptoměnové adresy. Slouží totiž jako zdroj dat o nahlášeních spojených s danou adresou a mohou obsahovat informace jako např. o jaký podvod se jedná, na jaké platformě se objevuje, kdo za ním stojí a také kdy byl nahlášen. Tyto informace pak většinou obohacuje také delší popis daného podvodu. Vzhledem k tomu, že ne každý zdroj obsahuje veškeré tyto informace, tak jsou data uchovávána v souborech. Soubory pak také umožňují informace zachovat v takové podobě, v jaké byly staženy, což umožňuje případnou kontrolu toho, zda tyto data nebyly na serveru upraveny.

Pro uchování souborů je využita složka `crawled_data`, ve které jsou jednotlivé soubory ukládány do podsložek, které mají názvy odvozené od identifikátorů URL podkategorií zdrojů. Dle názvu souboru pak lze jednoduše odvodit, jaký obsah se v něm nachází. Příkladem mohou být názvy – „3QkNpeRH89ZpofYbQ5gvckfBAZQ6V3yShp.html“ (soubor s nahlášeními k dané adrese), „reported_btc_addresses_1.html“ (soubor s první stranou BTC adres, které byly nahlášeny).

Soubory pak dále mohou sloužit jako záloha v případě, kdy by zdroj, ze kterého byly staženy, byl vypnut, ať už dočasně či trvale. Tato možnost rozhodně není pouze hypotetická, jelikož i při vývoji několikrát došlo k tomu, že byl některý ze zdrojů nedostupný.

4.4 Propojování dat s adresami

Po dokončení procházení konkrétní URL je vždy potřeba všechny adresy, které byly obsaženy na dané stránce či v jejím souboru, propojit se souborem, který je na serveru uložen. To pak umožňuje na webu vyhledávat adresy a načítat data s nimi spojená. Propojení páruje identifikátor adresy a dat do dvojice, u které lze také určit, jaké uživatelské role na webu či v API k těmto datům mají přístup. Omezení na straně propojení bylo místo omezení na straně dat zvoleno z toho důvodu, že lze takto vybrat konkrétní adresy, které z nějakého důvodu nechceme určité uživatelské roli zpřístupnit a stejně tak mohou být selektivně vybrány i některá data. Kdyby totiž byly omezení čistě na straně adresy či dat, pak by to velmi omezovalo právě možnosti volby. Změnu přístupu je nutné provádět v databázi, jelikož tato možnost nebyla z časových důvodů do aplikace implementována. Jedná se tak o něco, co by mohlo být přidáno v rámci možného rozšíření této práce.

4.5 Zjištění kryptoměny adresy

Po získání seznamu Bitcoinových adres a dokončení procházení zdrojů je na řadě zjištění kryptoměn adres, u nichž tato informace dosud chyběla. Tento problém se týká pouze ne-Bitcoinových adres, jelikož Bitcoinové adresy jsou v databázi všechny. Vzhledem k tomu, že není cílem daného řešení se zaměřovat na jiné kryptoměny než Bitcoin, tak by se mohlo zdát nepodstatné k jaké altcoinové kryptoměně daná adresa patří. Jelikož však zdroje, se kterými dané řešení pracuje, obsahují kromě Bitcoinových adres také velké množství altcoinových adres, tak je vzhledem k možnému rozšíření dané práce vhodné minimálně implementovat určitou formu zjištění. Tento způsob řešení je pak možné případně zefektivnit či přejít na úplně jiný způsob v případě, že by se systém rozšiřoval o práci i s jinými kryptoměnami než jen s Bitcoinem, kde by dávalo smysl opět získávat seznam adres přímo z příslušných blockchainů konkrétních kryptoměn.

Zjištění kryptoměny probíhá skrze odesílání požadavků na službu Blockchair, která obsahuje data z několika blockchainů současně. Do URL požadavku stačí zadat kryptoměnu a adresu např. takto: „blockchair.com/KRYPTOMĚNA/address/ADRESA“ a ze zdrojového kódu stránky lze poté získat informaci o tom, zda se na daném blockchainu adresa nachází či nikoliv.

Ne vždy musí být daná adresa součástí pouze jednoho blockchainu. U Bitcoinu se např. může stát, že daná adresa bude také součástí blockchainu kryptoměny Bitcoin Cash, jelikož její historie vychází ze stejného blockchainu. Tento problém je řešen pořadím zjišťovaných kryptoměn tak, že první kryptoměna, která je u dané adresy nalezena je k ní přiřazena a další zjištění již nenásleduje. U adres nacházejících se současně na obou blockchainech Bitcoinu a Bitcoin Cash to tedy znamená, že budou vždy považovány jako adresy Bitcoinu.

Díky tomuto řešení lze v databázi uchovávat unikátní seznam adres, které jsou zároveň vždy součástí právě jednoho blockchainu. Řešení to není ideální, jelikož by správně měly být k dané adrese přiřazeny všechny její kryptoměny, nicméně toto řešení bylo zvoleno z toho důvodu, že se práce zaměřuje primárně na Bitcoin. Případná podpora dalších kryptoměn je pak implementována pouze navíc a tak, aby nebylo u Bitcoinu přistupováno k žádným ústupkům. Dalším důvodem je pak to, že u dat, která jsou získána procházením zdrojů, není možné jednoduše určit, pro který blockchain jsou určeny. Při propojování dat s adresami by tak byly propojeny pravděpodobně s adresami na více blockchainech, což by bylo velmi nepřesné, jelikož adresa na jiném blockchainu nemusí být spojena s nahlášeným podvodem. Tato situace může sice nastat i nyní, nicméně je větší pravděpodobnost, že se bude nahlášení týkat přímo Bitcoinu, jelikož je tato síť oproti ostatním výrazně větší a to jak v počtu adres, tak transakcí mezi nimi.

Pokud by bylo cílem práci rozšířit i o další kryptoměny, pak by samozřejmě bylo nutné buď ustoupit od unikátního seznamu adres či povolit u dané adresy přiřazení více kryptoměn současně a vyřešení problému s propojením dat a adres.

4.6 Automatizovaný běh

Automatizovaný běh programu zajišťuje jeho nepřetržitý chod bez nutnosti zásahu ze strany uživatele. V hlavní smyčce programu je procházení zabaleno do try-catch bloku. Tento blok buď úspěšně proběhne a následně se čeká na opakování hlavní smyčky, nebo se v něm objeví nějaká výjimka a pak zobrazí chybovou hlášku a taktéž čeká. Doba čekání se odvíjí od nastavení v souboru `config.json` a může se u obou zmíněných situací lišit.

Samotný program pak nenabízí téměř žádnou interakci s uživatelem, jelikož jedinou změnu jeho chování je možné ovlivnit zadáním vstupních parametrů – resetování databáze, restartování databáze a smazání instalační konfigurace. Pokud dané parametry zadány nejsou, pak program neresetuje databázi, restartuje ji a nesmaže instalační konfiguraci. Po spuštění pak již uživatel nemá možnost do procesu vstupovat a může jej tak pouze ukončit. Mohlo by se zdát, že takovýto program by mohl běžet i bez konzole nicméně není tomu tak. I přestože do konzole není zadáván žádný vstup, tak je její význam značný, jelikož vypisuje důležité hlášky týkající se výjimek vyvolaných chybami nejen při procházení a zobrazuje také, zda byla hlavní smyčka dokončena úspěšně či nikoliv a také čas, kdy dojde k jejímu opakování. Tyto informace slouží k jednodušší identifikaci chyb a jejich následnému odstranění.

Program by mohl být dále rozšířen o uživatelský vstup, který by např. vybíral, které zdroje budou procházeny či dočasně vypnul zjišťování kryptoměn adres. Dále by pak mohl být vypisován průběžný postup procházení či dostupnost nových dat u jednotlivých zdrojů. Tyto funkce již nebyly do výsledného řešení implementovány ze snahy stihnout odevzdat práci v termínu jejího odevzdání a také proto, že na rozdíl od jiných implementovaných funkcí nejsou pro dané řešení zásadní.

Kapitola 5

Implementace webu

Web je nástroj umožňující využití databáze s údaji o kryptoměnových adresách. Tento nástroj je navržen v programovacím jazyce JavaScript a využívá běhové prostředí Node.js. Jeho cílem je převážně poskytovat přístup k informacím z databáze a to dle určité hierarchie uživatelských rolí. Jako vstupní vrátka k informacím pak slouží nejen webové stránky, ale také API. Zatímco webové stránky jsou určené pro snadné, přehledné a uživatelsky přívětivé použití, tak API cílí spíše na zapojení do aplikací 3. stran.

V rámci implementace webových stránek je nejprve popsán proces vyhledávání kryptoměnové adresy. V tomto procesu dochází k načítání a zpracování dat souvisejících s vyhledanou adresou a jejich následnému zobrazení na webu. V těchto datech se nachází informace o nahlášených podvodech tj. název podvodu, platforma na níž figuruje, datum a čas nahlášení a mnohdy také název jedince či skupiny, která za daným podvodem stojí. Ne vždy se však musí jednat o pravdivé informace, jelikož zdroje ze kterých je čerpáno umožňují podat nahlášení komukoliv, zdarma a zřejmě také bez jakékoliv moderace, takže je potřeba brát veškeré informace s rezervou. Tak či tak se jedná o informace, které je třeba uchovávat a využívat s cílem zvýšit povědomí o možných rizicích spojených s kryptoměnami.

Dále je zmíněno zobrazení seznamu všech kryptoměnových adres v databázi, k čemuž nástroj využívá samostatnou stránku. Tato stránka umožňuje adresy filtrovat dle vybrané kryptoměny a také zdroje, ke kterému se váže alespoň jedno její nahlášení. Jednotlivé adresy v seznamu pak obsahují odkaz na stránku s jejími informacemi stejně, jako by byla daná adresa vyhledána.

Součástí webových stránek jsou dále statistiky obsahující počty adres u jednotlivých zdrojů, které jsou navíc rozděleny dle příslušných kryptoměn. Vzhledem k časové náročnosti získání těchto statistik z databáze se tyto údaje aktualizují vždy jednou za daný časový interval a po vybranou dobu jsou pak načítány pouze z mezipaměti. Touto optimalizací se šetří nejen prostředky serveru, ale také čas uživatele, jelikož u tohoto typu aplikace není nutné v danou chvíli poskytovat skutečný počet adres v reálném čase.

Z pohledu uživatele je poté zásadní zmínit zapojení uživatelských účtů a proces jejich registrace a přihlášení, jelikož každý uživatelský účet vystupuje v nějaké uživatelské roli, což ovlivňuje k jakým datům má přístup. Účet pak navíc uživateli otevírá možnost k využití API. Při registraci se totiž generuje API tokenu, který je pro využití API vyžadován.

Z technického hlediska poté nesmí chybět zmínka o implementaci směrování požadavků pomocí Express.js a také generování statického obsahu stránek. Jednotlivé stránky pracují s určitým uživatelským prostředím, které je blíže představeno a je zmíněn jeho responzivní design a také tmavý režim.

Poslední zmíněnou částí webového nástroje je API. API poskytuje přístup k datům v databázi bez nutnosti použití již zmíněných webových stránek. Data jsou v rámci API poskytována ve formátu JSON a použití API je vhodné pro zapojení do aplikace 3. strany. Pro omezení přístupu k datům pro jednotlivé uživatele a zabránění potenciálnímu zneužití je pro použití API vyžadován token, který zaznamenává každý požadavek. Při dosažení limitu pro daný uživatelský účet je pak přístup k API pro daný token dočasně zablokován.

5.1 Vyhledávání kryptoměnové adresy

Nejdůležitější funkcí tohoto řešení je vyhledávání kryptoměnové adresy. Adresu lze na webu vyhledat jejím zadáním do vyhledávacího pole uprostřed (na úvodní stránce) či na vrchu obrazovky. Po potvrzení stisknutím klávesy [Enter] či odesláním formuláře se daný požadavek zpracuje a web se přesměruje na příslušnou URL. Na této URL se poté nachází údaje k samotné adrese.

Cílem vyhledávání je nejprve načíst informace z databáze a z disku, zpracovat je a poté poskytnout odpovídající výstup. V případě, že byla adresa nahlášena na jednom z používaných zdrojů, jejichž seznam je uveden v tabulce 2.1, a již byla tato informace získána crawlerem, tak se dané nahlášení zobrazí na webu. Lze tak jednoduše zjistit, zda daná adresa již byla na některém ze zdrojů nahlášena a to bez nutnosti jejich navštívení. Pro snadné a rychlé ověření existence nahlášení je u každého z nich uveden příslušný odkaz, ze kterého bylo dané nahlášení staženo.

Výstup může dále obsahovat odkazy na blockchainové průzkumníky a to za určitých pravidel, o nichž pojednává sekce 5.1.3. Blockchainový průzkumník obsahuje informace jako seznam transakcí, jejich vstupy, výstupy a také čas a částku, která byla přijata či odeslána. Tyto informace pak umožňují manuální kontrolu toho, zda by mohlo být některé z nahlášení oprávněné. Např. pokud někdo nahlásil danou adresu s tím, že na ní poslal 1 BTC a transakce o této částce je v seznamu transakcí obsažena, pak lze předpokládat, že dané nahlášení může být odůvodněné.

5.1.1 Načítání dat

Data o vybrané adrese jsou načítána z databáze a z disku. V databázi je uložena informace o příslušnosti adresy k některé z kryptoměn, datum získání posledních dat k dané adrese a seznam identifikátorů dat, které s danou adresou souvisí. Pomocí těchto identifikátorů jsou poté z databáze získány URL řetězce, ze kterých byly soubory staženy a také cesty k daným souborům na disku. Právě díky těmto cestám je možné následně načítat soubory a po jejich zpracování zobrazit odpovídající informace na webu.

5.1.2 Zpracování dat

Jak již bylo zmíněno v předchozí sekci 5.1.1, data jsou načítána ze souborů, které obsahují informace o dané adrese a byly staženy z některého z vybraných zdrojů. Mohlo by se zdát jako nejlepší řešení soubory načítat a zpracovávat již v crawleru a na webu pouze získávat odpovídající informace. Toto řešení je sice vhodnější, pokud je očekávána vyšší návštěvnost webu, jelikož není potřeba data zpracovávat pro každý jeden webový požadavek, nicméně také vyžaduje vyšší kapacitu disku, jelikož v podstatě dochází k duplikaci již lokálně dostupných dat. Od této možnosti bylo tedy nakonec upuštěno, vzhledem k omezeným prostředkům při vývoji, protože mimo jiné již uložení všech BTC adres z blockchainu zabírá několik desítek GB, a místo toho bylo zvoleno zpracování na webu.

U HTML souborů je zpracování založeno na vyhledání specifických elementů pomocí předem navržených CSS selektorů. Tyto selektory umožňují vybrat důležité elementy obsahující potřebné informace. Některé textové řetězce mohou obsahovat mezery na začátku či konci, což je ošetřeno využitím funkce `trim()`, která tyto mezery odebírá. U nahlášení se pak pracuje datem jeho odeslání, což je při zpracování problém, jelikož téměř každý zdroj používá jiný formát. Z toho důvodu byla vytvořena funkce `parseDateTime()`, která umí pracovat s datумы obsahujícími pomlčky, tečky, čárky, textovou formou měsíců a dokonce také čas. Po zpracování jsou tedy všechny tyto formáty zpracovány do jednoho objektu třídy `Date` a obsahují datum a čas nahlášení.

Zpracování souboru ve formátu JSON je o něco jednodušší jednak díky tomu, že jej používá pouze jeden zdroj a také díky tomu, že jsou data v něm uložena způsobileji. Z již zmíněných funkcí je tedy využívána pouze funkce `trim()`.

5.1.3 Informace k dané adrese

Po vyhledání kryptoměnové adresy nebo jejím vybrání v seznamu adres, po načtení a zpracování dat se na webu objeví informace spojené s touto adresou. Nejprve je zde pomocí obrázku znázorněna její příslušnost k určité kryptoměně. V případě, že se jedná o jednu z podporovaných kryptoměn na webu `blockchair.com` jako Bitcoin, Ethereum a další, pak jsou zde přítomny také odkazy na 2 blockchainové průzkumníky, na nichž se daná adresa nachází.

Prvním průzkumníkem je vždy `blockchair.com` a dalším ten, který podporuje danou kryptoměnu. Ne každý průzkumník totiž obsahuje více než jednu či dvě kryptoměny. Zároveň druhý průzkumník může fungovat jako alternativa v případě, kdy první zmíněná možnost není z nějakého důvodu dostupná. Pomocí blockchainového průzkumníka lze prohlížet vstupy, výstupy, částky a také čas všech transakcí a to u všech kryptoměn, které tyto informace poskytují veřejně. Tímto případem je také Bitcoin.

Podvodníci se obvykle snaží ztížit analýzu transakcí a dohledání cílové adresy. Využívají k tomu kryptoměny, které neposkytují veřejnou historii transakcí. Příkladem takové kryptoměny je Monero. Monero poskytuje přístup k historii transakcí pouze s pomocí privátního klíče, takže bez něj není možné transakce procházet.

Častěji se ovšem k podvodům využívá Bitcoin a u něj se pro ztížení analýzy používá kryptoměnových mixérů. Mixér funguje tak, že podvodník pošle z dané adresy určitou částku a to stejné udělají i ostatní aktéři v dané síti. Poté mixér provede spoustu transakcí na různé adresy a na cílovou adresu podvodníka dorazí o něco menší množství kryptoměny dle výše transakčních poplatků v dané síti.

Právě pro identifikaci mixérů je možné využít průzkumníky. Hodí se však také pro ověření legitimacy jednotlivých nahlášení. Jedním z problémů s těmito nahlášeními je, že sice mohou pomoci při odhalení podvodu, nicméně stránky na které jsou tyto podvody nahrávány jejich pravost neověřují a odeslat je může prakticky kdokoli. Pokud je však v nahlášení obsažena určitá částka, kterou daná oběť odeslala, či je uveden čas, kdy k dané transakci došlo, pak je možné tyto údaje v průzkumníkovi ověřit. Zahrnutí jednoho či více odkazů na průzkumníky také není mezi weby zabývajícími se nahlášeními nic nového a používá je např. služba `bitcoinabuse.com`.

Všechny dosud zmíněné informace k dané adrese se týkají všech adres v databázi, tedy i těch, které nejsou součástí žádného nahlášení. Web a také API lze tedy mimo jiné využít také k rychlé kontrole toho, zda je vyhledaná adresa obsažena na Bitcoinovém blockchainu, což je funkce, která není pro toto řešení zásadní, ale může být pro určité situace užitečná.

Nejdůležitější součástí stránky s informacemi k dané adrese je seznam nahlášení. Web poskytuje již zpracovaná data z databáze v podobě několika informačních položek ke každému nahlášení. Všechny možné informační položky jsou uvedené v tabulce 5.1. Nahlášení jsou poté pro lepší orientaci rozdělena do jednotlivých stránek po 20 nahlášeních. O stránkování nejen nahlášení, ale i jiného obsahu na webu však více pojednává sekce 5.1.6.

Jednou z žádaných funkcí řešení je možnost identifikace a deanonymizace vlastníků Bitcoinových adres. Případné možnosti v této oblasti jsou blíže představeny v sekci 5.1.4, která navíc s pomocí několika snímků z webu představuje možné řešení s využitím tohoto nástroje. To, zda je jakékoliv nahlášení z webu k těmto účelům vůbec možné použít a jak lze ověřit důvěryhodnost nahlášení přibližuje sekce 5.1.5.

| Typ | Popis |
|----------------|--|
| Datum | Datum a čas, kdy bylo dané nahlášení publikováno na použitém zdrojovém webu. |
| Typ | O jaký typ podvodu se jedná. Může být vybrán ze seznamu webem předem připravených možností či zadán ručně. |
| Platforma | Na jaké platformě se daný podvod objevil. |
| Země | Z jaké země nahlášení pochází. |
| URL | URL adresa na detail adresy na zdrojovém webu, kde je možné ověřit, že takové nahlášení skutečně existuje. |
| Podvodník | Jaký subjekt za daným podvodem stojí. |
| Popis | Delší popis nahlášení. Může obsahovat detailní informace o daném podvodu. |
| Chybová hláška | Objevuje se v případě, kdy adresa v databázi sice existuje, ale není spojena s žádným nahlášením. |

Tabulka 5.1: Typy informací obsažené v nahlášení kryptoměnové adresy

5.1.4 Deanonymizace vlastníka Bitcoinové adresy

Bitcoinová síť je pseudoanonymní, což znamená, že žádná Bitcoinová adresa není přímo spojena s jejím vlastníkem. Vlastníkem Bitcoinové adresy je subjekt, který má přístup k jejímu privátnímu klíči, který mu umožňuje podepisovat transakce a tedy odesílat určité množství bitcoinů na libovolnou adresu. To však vede k několika možným scénářům:

- Existuje více subjektů a každý zná celý privátní klíč. Je tedy **více vlastníků adresy** a každý z nich může provádět transakce.
- Existuje pouze jeden subjekt, který zná celý privátní klíč a je tedy **jediným vlastníkem adresy**.
- Existuje více subjektů, kdy každý zná pouze část privátního klíče, ale ani jeden nezná celý privátní klíč. Je třeba spojit všechny části privátního klíče, aby bylo možné autorizovat transakce a tedy **nikdo de facto není vlastníkem**.
- Neexistuje ani jeden subjekt, který by znal celý privátní klíč a nelze ani spojit několik částí klíče od jednotlivých subjektů. Odhaduje se, že přibližně 20% všech bitcoinů je obsaženo na adresách k nimž již **nikdo nemá přístup**. [19]

Jak je z výše uvedených možností patrné, vlastnictví Bitcoinové adresy nelze jednoznačně určit. Dalo by se namítnout, že lze vlastníka adresy odhalit právě tím, že v určitý moment provede transakci a tedy zaručeně v daný moment je jejím vlastníkem. Toto tvrzení je pravdivé pouze za předpokladu, kdy máme stoprocentní jistotu, že tuto transakci skutečně provedla daná osoba např. pomocí hardwarové peněženky. Problémem tohoto ověření je pak také fakt, že daný subjekt může po provedení transakce privátní klíč ztratit či se jej zbavit, čímž daná adresa přijde o jejího vlastníka.

Nelze ani očekávat, že by privátní klíč daný subjekt zveřejnil, jelikož by tím umožnil přístup k dané adrese komukoliv, kdo by pomocí něj mohl veškeré bitcoiny poslat na vlastní adresu.

Při opuštění vlastníků privátních klíčů se dá na danou problematiku podívat ještě z jednoho pohledu. Pro příklad si lze představit následující situaci. Podvodník si založí účet např. na Twitteru či jiné sociální síti a na této síti zveřejní svou adresu s cílem vylákat od obětí kryptoměny. Aby daný podvod byl alespoň trochu uvěřitelný, tak k této adrese přiloží ještě zprávu o tom, že po odeslání kryptoměny se uživateli vrátí její dvojnásobek. Prohledáme v tomto případě Twitter, nalezneme Bitcoinovou adresu v příspěvku od tohoto podvodníka a známe vlastníka. Takto jednoduché to však není. Jak zjistíme, že ten kdo daný příspěvek odeslal skutečně vlastní její privátní klíč? Jak poznáme, že skutečný vlastník privátního klíče nepoužil bílého koně, tedy jiný subjekt, který tento příspěvek publikoval na sociální síti za něj? Podvodník také mohl např. využít bota pro vytvoření několika různých účtů, které by zveřejnily různé podvody s touto adresou. Na sociální síti navíc může figurovat subjekt, který se snaží na podobné podvody upozornit (např. policie) a i on může publikovat danou adresu, samozřejmě s opačným cílem. Z toho tedy vyplývá, že je víceméně irrelevantní, kdo danou adresu zveřejňuje, jelikož není možné určit, zda se skutečně jedná o vlastníka dané adresy. Prohledávání sociálních sítí s cílem propojit určité účty s Bitcoinovými adresami tedy může pomoci identifikovat další místa na internetu, kde se daná adresa vyskytuje a v omezeném případě dokonce také může vést k identifikaci vlastníka adresy. Využití takovéto funkce je nicméně omezeno na specifické případy, kdy osoba na sociální síti vystupuje pod skutečným jménem a následně je možné vlastnictví ověřit tak, že daná osoba prokazatelně odešle transakci z dané adresy, čímž se vlastnictví ověří. Jak již bylo zmíněno tato kontrola je velmi individuální a bez ní k deanonymizaci nemůže dojít. Samotné využití obecného prohledávání sociálních sítí je tedy velmi minimální a neefektivní s ohledem na vynaložené prostředky. Efektivněji lze prohledávání sociální sítě využít pouze při zaměření na osoby u nichž již víme, že danou adresu vlastní. V této situaci však již pracujeme s deanonymizovaným vlastníkem adresy, takže je tento krok zbytečný.


Od deanonymizace ovšem nebylo v této práci upuštěno úplně. Díky tomu, že tato práce pracuje s uživatelskými nahlášeními adres, tak je možné občas najít nahlášení, u nichž je nějaká osobní informace obsažena. Příklady takovýchto nahlášení je možné vidět na obrázcích 5.1 a 5.2. Využitelnost takovéto informace potom záleží na důvěryhodnosti daného nahlášení a rozhodně nelze tvrdit, že by jakákoliv informace musela být pravdivá. Určitým způsobem však lze alespoň částečně ověřit pravdivost určitých informací, o čemž blíže pojednává následující sekce 5.1.5.

Proč bylo tedy upuštěno od vyhledávání na sociálních sítích a místo toho byly použity uživatelské nahlášení, když ani u nich není zaručena deanonymizace? Protože je cílem nejen deanonymizovat vlastníka, ale také přinést přidanou hodnotu v podobě upozornění na možné podvody.

Vyvíjené řešení tedy nepostrádá deanonymizační prvky úplně, jelikož lze narazit na nahlášení obsahující osobní informace, ale jejich využití je více dedikováno na samotného uživatele a jeho následnou analýzu. Individuální analýza s cílem odhalit vlastníka adresy je efektivnější a také spolehlivější než jakékoliv automatizované řešení pracující s informacemi dostupnými na internetu.

Address

Address

 12qusMrNaac75kHFWMKPQuuSn7L44xozvV

Blockchain explorers

[View on blockchain.com](#)

[View on blockchair.com](#)

Last update

25. 3. 2023 18:10:35

| Date | Type | Country | URL |
|---------------------|------------|---------------|---------------------------------------|
| 3. 3. 2023 19:22:59 | ransomware | United States | View on cryptscam.com |

Abuser

Recover

Description

Good work deserves recommendation... i lost over 2.3 BTC on Instagram bitcoin scam.. Right about 2 weeks after my ordeal with them I tried using the recommendation from someone on one of the comment section (www.ukmcybersecurity.com) I was able to get all my money back in less than 48 hours.. Contact (www.ukmcybersecurity.com) to recover all your stolen bitcoins free of charge

| Date | Type | Country | URL |
|---------------------|-------|---------------|---------------------------------------|
| 3. 3. 2023 18:57:07 | other | United States | View on cryptscam.com |

Abuser

Michael Wisard


Description

This man was going to help me get back some money from scam, he promised to pay me back toda yesterday and the day before that, but he started to ask for more and more fees, I even had to ask my sister to help me and she paid 800 euro in total. He has used 2 walletaddresses wich I sent to. From the date 22 februari-23. After the last sending this morning and he was supposed to send the bitcoin, he called and told me that I had to pay 550 euro more fir autorisationfee whatever that is. I asked him to pay everything back but he say he can because the funds are locked in the wallet. I want him to send back everything to me. But he refuse. Can you assisist on this. I guess this upfront fees are scams.

Obrázek 5.1: Příklad nahlášení obsahujícího jméno

Address

Address

 17WqsBwFEUij1Tgu2RMiL1wFRq6kRhGUNq

Blockchain explorers

[View on blockchain.com](#)

[View on blockchair.com](#)

Last update

25. 3. 2023 18:22:01

| Date | Type | Country | URL |
|-------------------------|---------------------|---------------|---|
| 10. 1. 2023 22:53:48 | Drug dealer scammer | United States | View on checkbitcoinaddress.com |

Abuser

Homamdan

Description

User is on telegram by the name of homamdan .

Obrázek 5.2: Příklad nahlášení obsahujícího přezdívku

5.1.5 Důvěryhodnost nahlášení

Nedílnou součástí stránky s informacemi k vyhledané adrese jsou uživatelská nahlášení. Nahlášení jsou obsažena na zdrojových webech a stažena pomocí crawleru. Tyto weby umožňují téměř komukoliv nahlásit podvod spojený s kryptoměnami a obvykle umožňují kromě Bitcoinových adres nahlásit i adresy jiných kryptoměn. Výhodou těchto služeb je, že pomocí jednoduchého formuláře umožňují rychle upozornit na danou hrozbu. Na druhou stranu však nahlášení nejsou na těchto webech žádným způsobem moderována a tedy mohou obsahovat zavádějící informace. Motivů k odeslání zbytečného, nepravdivého či škodlivého nahlášení může být více. Podvodníci se např. mohou snažit zahltit tyto služby falešnými zprávami, aby mezi nimi nebylo snadné nalézt nahlášení jejich vlastního podvodu. V horším případě pak mohou zneužít nahlášení jiného podvodu s cílem propagovat podvod vlastní a teoreticky tak zneužít danou platformu k opačnému účelu, než k jakému byla zamýšlena. Zahlcením lze také cílit na možnou demotivaci uživatele danou službu používat.

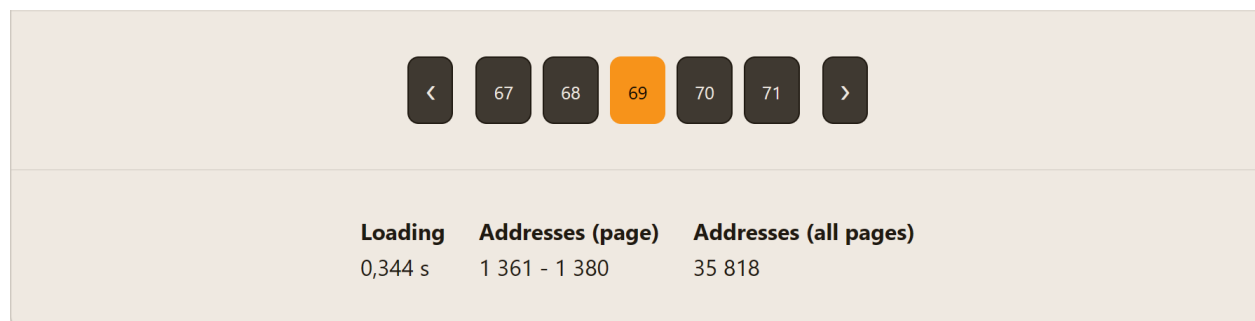
Nelze však říci, že by na daných platformách neexistovaly žádné smysluplné nahlášení. Z pohledu oběti může být často náročné správně identifikovat situaci a všechny okolnosti podvodu a tedy nemusí být podvod vždy správně a dostatečně podrobně popsán. Nicméně jakékoliv nahlášení, které u dané adresy je, může vést ke zvýšené obezřetnosti uživatele.

Detailně popsané nahlášení obsahující místo, průběh a jiné další okolnosti podvodu mohou uživatelům pomoci se takovému podvodu vyhnout. Pokud pak navíc nahlášení obsahuje jakékoliv informace o podvodníkovi, ať je to jeho jméno, přezdívká, telefon, email, účet na sociální síti či jazyk použitý při komunikaci, tak tyto informace mohou být použity při pátrání po dané osobě.

5.1.6 Stránkování obsahu

Stránkování obsahu slouží k rozdělení seznamu dat po určitém počtu položek na stránku tak, aby byl seznam přehledný, rychle se načítal z databáze a aby bylo snadné jeho procházení. Počet položek by tedy neměl být příliš vysoký, aby většina informací byla viditelná v rámci jedné obrazovky. Pro všechny stránky využívající stránkování obsahu byl zvolen jednotný počet 20 položek na stránku. Tento počet je však možné individuálně nastavit v souboru `config.json`. Stránkování je využíváno u všech seznamů zobrazovaných na webu. Konkrétně se jedná o seznam nahlášení u vyhledané adresy, seznam adres a seznam uživatelů ve správě uživatelů. U posledních dvou je potřeba zmínit, že filtrace dat nijak zásadně neovlivňuje stránkování, jelikož jsou data rozdělena již při načítání z databáze. Vždy tedy bude na stránce zobrazen stejný počet položek. Jedinou výjimkou je poslední stránka, která obsahuje zbytkové položky po rozdělení celého seznamu a může obsahovat o něco méně položek, než jaký je stanovený limit.

V databázi je stránkování řešeno pomocí příkazů `LIMIT` a `OFFSET`. První příkaz ovlivňuje maximální počet získaných položek a druhý určuje počáteční položku výběru.



Obrázek 5.3: Navigace u stránkování obsahu








Na webu je stránkování doplněno o navigační tlačítka, které umožňují přepínat na jednotlivé stránky. Tlačítek s čísly stránek je vždy maximálně pět a jsou doplněna o tlačítka, které odkazují na první a poslední stránku. Pro informaci je zde uveden také celkový počet položek napříč všemi stránkami a konkrétní rozsah položek na aktuální stránce.

5.2 Seznam adres

Další důležitou funkcí webu je zobrazení seznamu nahlášených adres. Tento seznam obsahuje všechny kryptoměnové adresy, ke kterým má aktuální uživatel přístup a které jsou v databázi spojené alespoň s jedním staženým souborem. Samotnému propojování dat a adres se blíže věnuje sekce 4.4 kapitoly zabývající se implementací crawleru.

Seznam adres je seřazen sestupně dle data a času poslední aktualizace. Nejdříve se tedy v seznamu objeví naposledy aktualizované adresy. Datum poslední aktualizace je pak určeno nejnovějším staženým souborem spojeným s danou adresou. Toto seřazení bylo zvoleno záměrně a to proto, aby aktuální informace byly zobrazeny co možná neblíže počátku seznamu, jelikož běžný uživatel spíše využije aktuálnější informace než ty, které byly získány při prvním spuštění crawleru.

U jednotlivých adres je pak zobrazena ikona kryptoměny, ke které daná adresa patří. Je tak možné jednoduše identifikovat příslušnost k určité kryptoměně, bez nutnosti přecházet na stránku s danou adresou. Tento přechod však není nikterak náročný, jelikož všechny adresy obsahují odkaz, který vede na stejnou stránku, která je zobrazena po vyhledání adresy. Není tedy nutné využívat vyhledávač, i když i ten je na stránce se seznamem adres obsažen.

| Addresses | | |
|--|---------------------|---------------------|
| Currency | Bitcoin (BTC) ▼ | Source All ▼ Filter |
| Address | Last update | |
|  1KayqtCZJTVYPQjjqJHBdrtvE69DgW5xPL | 27. 3. 2023 1:54:15 | |
|  1AZSbcgaP7nfcdbXW2Qh66sFX13sPPfUt | 27. 3. 2023 1:09:55 | |
|  3GsNkaff4su5Tzbt8MFy6cS7rumWczrX9R | 27. 3. 2023 1:09:55 | |
|  1N7eMF7KnP8vQqHtn89dVPcfXqQXFkra6H | 27. 3. 2023 1:09:55 | |
|  123iEhDDGSNq6cJuRStmzzqe7yevkS1V2r | 27. 3. 2023 1:09:55 | |
|  1LTdSkZSFM2rd2XEMW8oaRrY3ZPtNzNiv3 | 27. 3. 2023 1:09:55 | |
|  1LAcPK26AvKWVge7q4u3zD71E7xZoPp3yB | 27. 3. 2023 1:09:55 | |

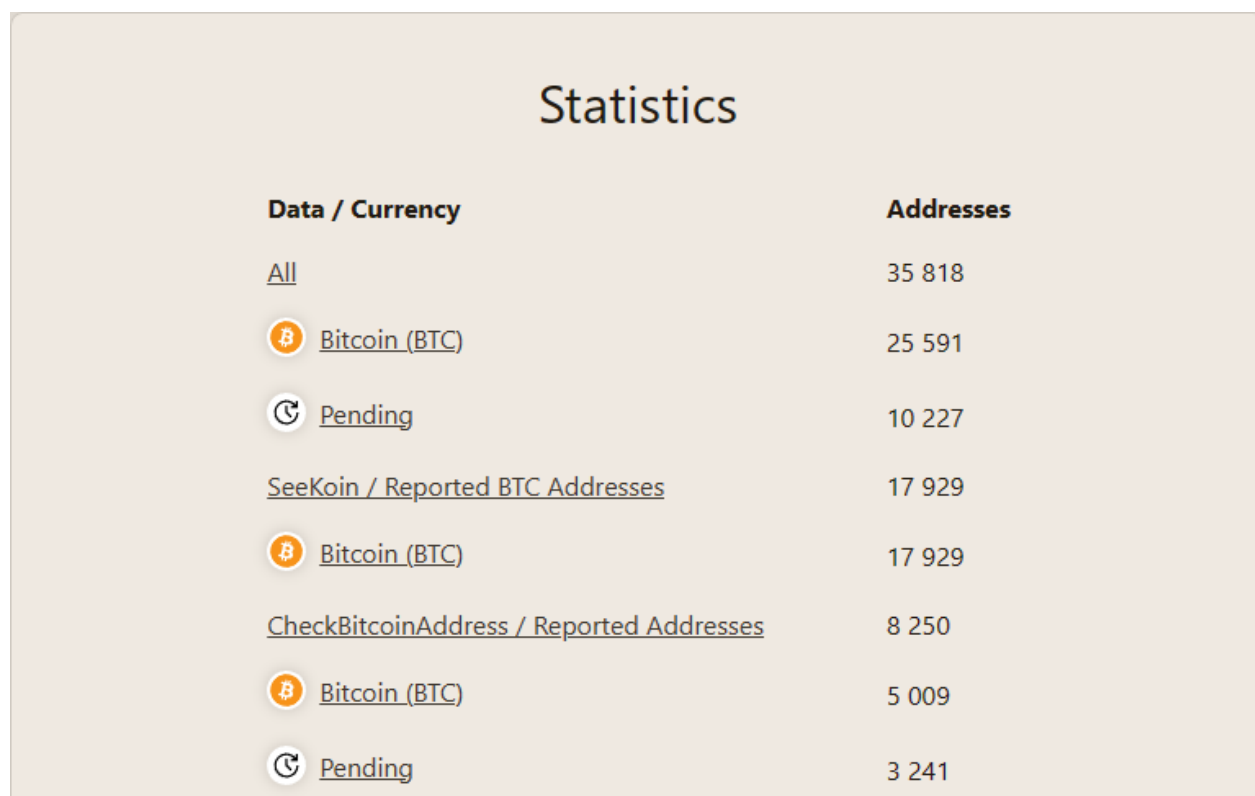
Obrázek 5.4: Ukázka části seznamu BTC adres






5.2.1 Filtrování seznamu adres

Jednou z klíčových funkcí seznamu adres je možnost jeho filtrování. Bez výběru filtru je ve výchozím stavu zobrazen seznam všech adres napříč všemi zdroji. Pokud je u kryptoměny vybrán např. Bitcoin a u výběru zdroje je ponechána možnost „All“, pak jsou zobrazeny všechny Bitcoinové adresy bez ohledu na zdroj. V případě, že je u kryptoměny ponechána tato možnost a navíc je vybrán zdroj, např. „CryptoBlacklist“, tak je zobrazen seznam BTC adres v tomto zdroji. Pokud je cílem získat adresy libovolných kryptoměn tohoto zdroje, pak stačí u kryptoměny zvolit možnost „All“.

5.3 Statistiky

Další funkcí webu je zobrazení souhrnných statistik nahlášených adres jednotlivých zdrojů. Tyto statistiky jsou členěny nejen dle zdroje, ale také dle jeho podkategorií. U každého celku jsou pak ještě rozděleny na jednotlivé kryptoměny. Všechny tyto informace jsou sice dostupné na stránce se seznamem adres po vybrání specifických filtrů, nicméně na této stránce jsou tyto informace zobrazené přehledně na jednom místě. Zdroje jsou seřazené sestupně dle počtu adres a celkový počet adres je možné najít úplně na prvním řádku. Příklad toho, jak mohou statistiky vypadat, je možné vidět na obrázku 5.5 níže.



| Data / Currency | Addresses |
|---|-----------|
| All | 35 818 |
|  Bitcoin (BTC) | 25 591 |
|  Pending | 10 227 |
| SeeKoin / Reported BTC Addresses | 17 929 |
|  Bitcoin (BTC) | 17 929 |
| CheckBitcoinAddress / Reported Addresses | 8 250 |
|  Bitcoin (BTC) | 5 009 |
|  Pending | 3 241 |

Obrázek 5.5: Ukázka části statistik

5.3.1 Ukládání do mezipaměti

I přes veškeré optimalizace a využití indexů je získání počtů adres z databáze stále nejpomalejší součástí webu. Stránka jejíž část je uvedena na obrázku výše se při testování načítala přibližně 7 až 8 sekund. Mohlo by se zdát, že tato doba není příliš dlouhá, nicméně její délka se odvíjí od počtu adres. Dá se očekávat, že při delším používání crawleru (uvedené statistiky odpovídají přibližně 6 hodinám spuštěného crawleru) bude počet nahlášených adres stoupat a tedy i čas na získání těchto statistik. Vzhledem k tomu, že zobrazení statistik není jednou z nejzásadnějších funkcí webu, na jejichž aktuálnosti by byla jakákoliv součást aplikace závislá, tak lze dlouhá doba načítání vyřešit následovně. Po prvním načtení se statistiky uloží do mezipaměti, ze které jsou následně po určité době získávány. Tímto způsobem se nejen výrazně zrychlí používání statistik, ale také sníží nároky na databázový server, jelikož počítání adres je také výpočetně náročný proces.

5.4 Uživatelské účty

Uživatelské účty poskytují příležitost pro vydefinování hierarchie uživatelů, což ovlivňuje k jakým datům mají uživatelé v aplikaci přístup, jaké omezení mají v rámci API a jaké další funkce mohou používat.

5.4.1 Registrace

Registrace probíhá skrze registrační formulář (viz obrázek 5.6), který vyžaduje zadání validní emailové adresy, alespoň 8místného hesla obsahujícího písmena (anglické abecedy), čísla a speciální znaky a jeho opětovné zadání. V případě, že daná emailová adresa v databázi dosud neexistuje, tak je účet úspěšně vytvořen a uživatel je automaticky rovnou i přihlášen a přesměrován na stránku s informacemi k jeho účtu. V opačném případě je zobrazena odpovídající chybová hláška. Chybové hlášky jsou zobrazovány také i pro libovolné dílčí nedostatky v rámci daného formuláře.

Pro šifrování hesla se používá balíček `bcrypt`. Pomocí jeho funkce `hashSync()` se pro dané heslo vytvoří hash, který je následně uložen do databáze.

Uživateli je při registraci přidělena uživatelská role. V případě, že se registruje jako úplně první v daném systému, tak je mu přidělena nejvyšší, administrátorská role. Očekává se tedy, že po zprovoznění webu se provozovatel jako první registruje, aby se náhodou administrátorem nestal jiný uživatel webu. Při registraci je pak uživateli vygenerován API token, který mu umožňuje API používat. O tom jak se od sebe jednotlivé role liší a kdo má jaké možnosti a limity pojednává sekce 5.4.3 zaměřená přímo na uživatelské role.

V registračním formuláři se navíc objevuje odkaz na přihlášení, které umožňuje snadné přesměrování pro uživatele, kteří již mají účet a chtějí se místo registrace přihlásit.

Sign up

Email

Password

Confirm password

Sign up

Already have an account?
[Sign in](#)

Sign in

Email

Password

Sign in

Do not have an account?
[Sign up](#)

Obrázek 5.6: Registrační a přihlašovací formulář

5.4.2 Přihlášení

Přihlášení probíhá skrze přihlašovací formulář (viz obrázek 5.6), který vyžaduje zadání validní emailové adresy a hesla. Validita se u přihlášení i registrace detekuje jak na úrovni HTML, tak na úrovni serveru, který zároveň rozhoduje o tom, jaký bude výsledek odeslaného formuláře. Aby bylo přihlášení úspěšné, tak se musí shodovat zadaná emailová adresa i heslo s údaji vyplněnými při registraci.

Již bylo zmíněno využití balíčku `bcrypt`. Tento balíček obsahuje také funkci `compareSync()`, která umožňuje porovnat zadané heslo s hashem z databáze, díky čemuž pozná, že se jedná o to stejné heslo. Využití hashování je pro zabezpečení hesla důležité, jelikož zamezuje jeho jednoduchému zneužití komukoliv, kdo má k dané databázi přístup.

Stejně jako u registračního formuláře je odkaz na přihlášení, tak i u přihlašovacího formuláře je odkaz na registraci. Cílem těchto odkazů je nabídnout uživateli rychlé přepnutí na odpovídající formulář.

5.4.3 Uživatelské role

Jak již bylo v této práci několikrát zmíněno, uživatelské role ovlivňují především to, jaké data bude mít uživatel dostupné a jaké limity bude mít při používání API. Nejméně možností má nepřihlášený uživatel, který nemá přístup k API a na webu má přístup pouze k datům určeným pro návštěvníka. Nejvíce možností má pak administrátor, který má kromě velmi vstřícného limitu použití API také možnost přistupovat k datům určeným specificky pro tuto roli, u kterých se neočekává žádné omezení. Navíc pak jako jediný může spravovat uživatele a upravovat jejich roli. Veškeré informace týkající se uživatelských rolí jsou dále shrnuty ve dvou následujících tabulkách 5.2 a 5.3.

| Typ uživatele | Použití API | Správa uživatelů | Přístup k datům určeným pro |
|-----------------------|-------------|------------------|-----------------------------|
| Nepřihlášený uživatel | NE | NE | Návštěvníka |
| Návštěvník | ANO | NE | Návštěvníka |
| Uživatel | ANO | NE | Uživatele |
| Insider | ANO | NE | Insidera |
| Administrátor | ANO | ANO | Administrátora |

Tabulka 5.2: Možnosti jednotlivých uživatelských rolí

| Role | Max. počet požadavků | Obnovení počtu požadavků po dosažení limitu za | Průměrný povolený počet požadavků za sekundu |
|---------------|----------------------|--|--|
| Návštěvník | 100 | 1 den | $1/864$ |
| Uživatel | 3 600 | 1 hodinu | 1 |
| Insider | 1 800 | 10 minut | 3 |
| Administrátor | 300 | 60 sekund | 5 |

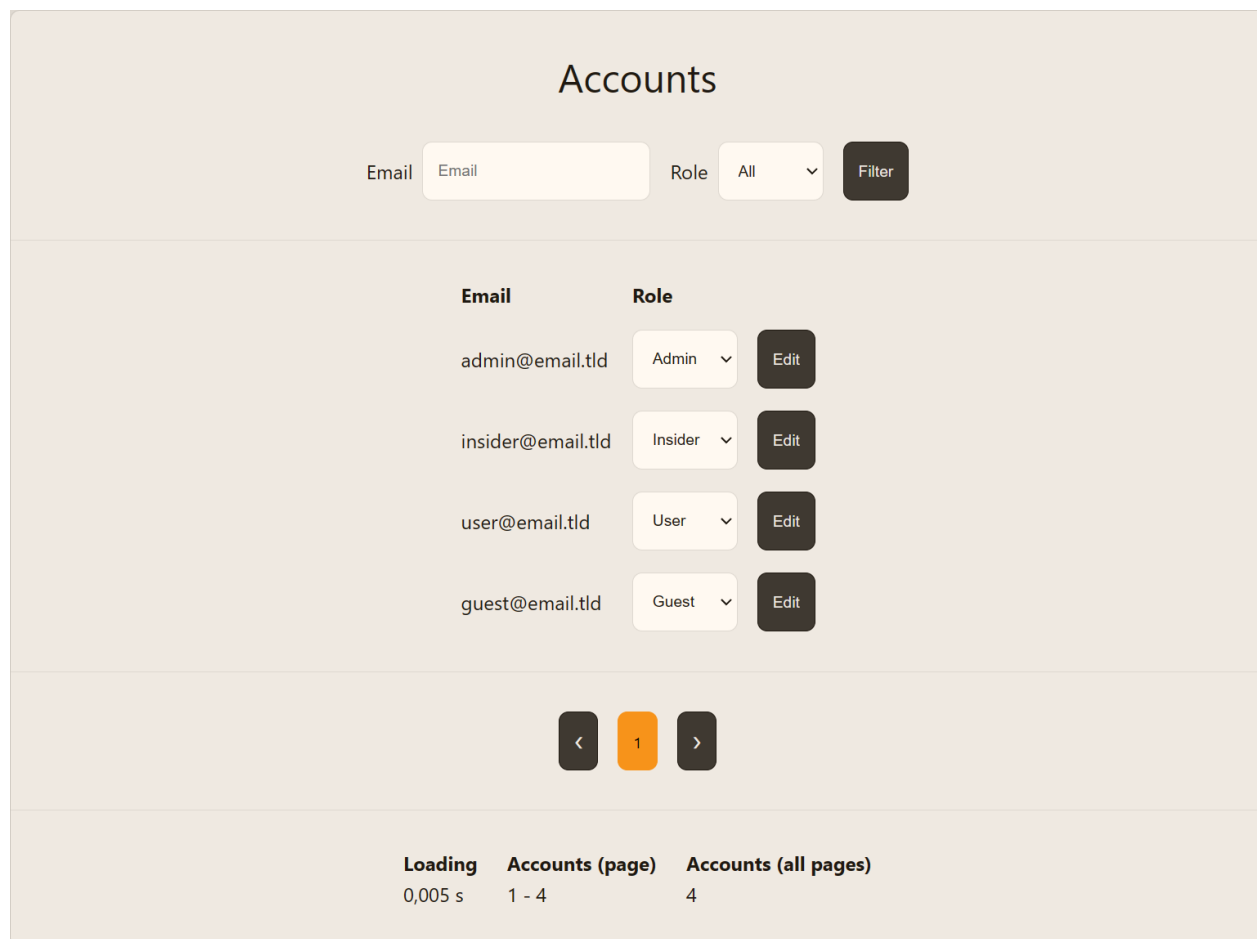
Tabulka 5.3: Limity počtu požadavků na API pro jednotlivé uživatelské role

5.4.4 Správa uživatelů

Administrátor je jediná uživatelská role, která má po přihlášení na webu přístup ke správě uživatelů. Po přihlášení uživatele se, stejně jako u jiných rolí, objeví stránka s informacemi o daném účtu. Na této stránce je pak kromě možnosti odhlášení navíc ještě možnost, která odkazuje právě na již zmíněnou správu uživatelů.

Hlavní funkcí na stránce se správou uživatelů je možnost upravit uživatelskou roli jednotlivých uživatelů. Administrátor může uživateli přidělit jednu ze 4 následujících rolí – návštěvník, uživatel, insider a administrátor. Cílem této volby je upravit omezení použití aplikace pro daného uživatele. Tato volba ovlivňuje nejen celkový přístup k datům, které se mezi jednotlivými rolemi liší, ale také frekvenci požadavků na API. V případě, že je danému uživateli přidělena role administrátora, pak je mu taktéž umožněno spravovat uživatele.

Na této stránce existuje speciální situace, kdy změna uživatelské role umožněna není a to v případě, kdy je aktuální uživatel posledním administrátorem v systému a pokusí se změnit svou vlastní roli. V takovém případě by tedy v systému nezůstal žádný administrátor, který by měl přístup k této správě, což je nežádoucí stav. Pokud je však v systému ještě další administrátor, pak toto povoleno je.



Obrázek 5.7: Správa uživatelů

Jak je z obrázku 5.7 patrné, tato stránka využívá některé společné prvky s jinými stránkami, jako je filtrace seznamu uživatelů a jeho rozdělení na několik stránek. Filtrování umožňuje omezit zobrazený seznam uživatelů pouze na uživatele vybrané uživatelské role. Po zadání emailové adresy je pak možné účet s danou adresou vyhledávat buď mezi všemi uživateli či opět pouze u vybrané skupiny uživatelů. Stránkování pak funguje podobně jako u seznamu adres či seznamu nahlášení a rozděluje uživatele po 20 na jednu stránku.

5.5 API

API je poslední důležitou funkcí webového nástroje, která byla navíc implementována nad rámec zadání. Umožňuje přístup k databázi adres a nahlášení bez nutnosti použití webového prohlížeče a je vhodné především pro implementaci do aplikací třetích stran.

Přístup k datům je stejně jako na webu řízen uživatelskými rolemi. Každý uživatel vlastní jeden API token, který ovlivňuje počet požadavků, které je schopen provést za určitý čas (viz tabulka 5.3). Tento limit nijak neovlivňuje ostatní uživatele. Token musí být umístěn v každém odeslaném požadavku a to způsobem, který je uveden v tabulce níže.

5.5.1 Funkce API

Následující tabulka 5.4 obsahuje všechny funkce API. U každé funkce je nejprve zmíněna relativní adresa, na kterou je potřeba požadavek odeslat. Pomocí velkých písmen jsou v této adrese zapsány parametry, které je možné do této adresy vložit. Tučně jsou zvýrazněny ty parametry, které jsou pro získání informací vyžadovány, zatímco kurzívou jsou uvedeny čistě volitelné parametry. Dále tabulka zmiňuje informace, které je možné při přijetí odpovědi očekávat.

| Relativní adresa požadavku | Očekávané informace |
|--|--|
| <code>/api/address/ADDRESS?token=TOKEN</code> | Obsahuje informace o dané adrese – příslušná kryptoměna, seznam identifikátorů datových souborů a datum poslední aktualizace. |
| <code>/api/addresses?token=TOKEN &page=<i>PAGE</i>&currency=<i>CURRENCY</i> &source=<i>SOURCE</i></code> | Obsahuje seznam nahlášených adres. U každé je uvedena příslušná kryptoměna, seznam identifikátorů datových souborů a datum poslední aktualizace. |
| <code>/api/currencies?token=TOKEN</code> | Obsahuje seznam kryptoměn. U každé je uveden její název a zkrácené označení (např. BTC). |
| <code>/api/data/DATA?token=TOKEN</code> | Obsahuje informace o datovém souboru – URL, ze které byl získán, délku jeho obsahu, datum získání a samotný jeho obsah. |
| <code>/api/sources?token=TOKEN</code> | Obsahuje seznam zdrojů. U každého je uveden identifikátor a název. |
| <code>/api/tokens/TOKEN</code> | Obsahuje informace týkající se API tokenu – počet použití, datum vytvoření, datum posledního použití a datum obnovení počtu použití. |

Tabulka 5.4: Seznam funkcí API

5.5.2 Parametry požadavků API

Jak již bylo zmíněno v předešlé sekci 5.5.1, adresy na které jsou odesílány požadavky obsahují určité parametry. Tyto parametry označují místa, která mají být nahrazena určitým řetězcem znaků. Některé řetězce je možné získat použitím API, jako třeba zkrácené označení kryptoměny či identifikátor zdroje a jiné jako třeba API token je nutné získat přímo z webu. Pomocí parametrů je možné s API pracovat podobně jako s webovou aplikací a např. u seznamu adres lze také filtrovat dle kryptoměny či zdroje. Snahou je tedy zachovat většinu funkcí, ale navíc nabídnout přístup také pomocí tohoto rozhraní. Rozhraní pak může sloužit při napojení stávajícího řešení do externí aplikace využívající některou z nabízených funkcí.

V tabulce 5.5 jsou uvedeny parametry požadavků API, které byly zmíněny v předchozí sekci 5.5.1. U každého z nich je pak uvedena její očekávaná hodnota a také místo, kde je možné ji nalézt.

| Parametr | Očekávaná hodnota | Kde lze nalézt požadovanou hodnotu |
|----------|---|---|
| ADDRESS | Kryptoměnová adresa | Je možné ji nalézt v seznamu adres na webu či v rámci API. |
| CURRENCY | Zkrácené označení kryptoměny (např. „BTC“) | Lze nalézt na webu na stránce k API či při získání seznamu kryptoměn v rámci API. |
| DATA | Číselný identifikátor datového souboru | Je obsažen v seznamu identifikátorů datových souborů u informací k dané adrese v rámci API. |
| PAGE | Číslo stránky | Na webu na stránce se seznamem adres je číslo stránky označeno v navigaci u příslušného tlačítka. |
| SOURCE | Číselný identifikátor zdroje | Nachází se v seznamu zdrojů v rámci API. |
| TOKEN | API token vygenerovaný při registraci uživatele na webu | Je možné jej nalézt v uživatelském účtu po přihlášení. |

Tabulka 5.5: Seznam parametrů v požadavcích API

Kapitola 6

Použité technologie

6.1 PostgreSQL

PostgreSQL je open-source objektově-relační databázový systém, který používá a rozšiřuje jazyk SQL o další funkce, které zlepšují škálovatelnost a bezpečné ukládání dat. Tento systém je spolehlivý, robustní a výkonný, což je také hlavní důvod, proč byl pro tuto práci zvolen oproti jiným alternativám. [20]

Hlavní výhodou PostgreSQL je jeho rychlost ukládání dat, což umožňuje zkrátit celkový čas potřebný pro crawlování externího obsahu, jelikož při crawlování je do databáze ukládáno velké množství dat. Snaha data co nejrychleji uložit do databáze má smysl z toho důvodu, že by data z externích zdrojů mohla být odebrána či by tyto zdroje nemusely být do budoucna dostupné. Čím dříve se tedy dostanou do databáze, tím dříve je zajištěno jejich uchování a také jejich použití v rámci vyvíjeného řešení. [21]

Rychlost čtení je pak o něco méně důležitá, jelikož toto řešení nevyžaduje použití real-time dat, které jsou navíc pro většinu uživatelů stejné. Uživatel tedy nemusí nutně pracovat s aktuální reprezentací dat v databázi a tyto data může načítat např. z mezipaměti serveru, čímž se snižuje význam rychlého čtení z databáze. [21]

Vyvíjená databáze používá PostgreSQL ve verzi 15.2 a k této databázi přistupují všechny nástroje řešení skrze různé rozhraní balíčků knihoven jednotlivých programovacích jazyků. Crawler, který je napsán v jazyce Python k tomu používá knihovny: `psycopg` a `psycopg_pool` a web společně s API pak Node.js balíčky: `pg` a `connect-pg-simple`.

6.2 Node.js

Node.js je open-source, multiplatformní běhové prostředí JavaScriptu. Jedná se o asynchronní událostmi řízený běhový modul JavaScriptu navržený k vytváření škálovatelných síťových aplikací. Node.js nepoužívá vlákna OS ani blokování procesu, jelikož téměř žádná funkce v Node.js přímo neprovádí vstupně-výstupní operace, což je v kontrastu k dnešnímu běžnějšímu modelu souběžnosti. [22]

Node.js obsahuje V8 JavaScript engine, stejný jako v Google Chrome a dalších prohlížečích. Je napsán v C++ a je schopen běžet na systémech macOS, Linux, Windows a dalších. Engine kompiluje JavaScriptový kód pomocí JIT (just-in-time), čímž urychluje jeho spuštění. [23]

Ve vyvíjeném řešení je použité prostředí Node.js ve verzi 18.15 LTS (verze s dlouhodobou podporou), kde je s pomocí webového frameworku Express.js vytvořena API a navíc ještě s pomocí šablonovacího jazyka EJS také web. Node.js tedy poskytuje prostředí pro zpracování HTTP požadavků obou nástrojů, k nimž zároveň poskytuje odpovídající obsah v podobě HTML (web) či JSON (API).

6.3 Express.js

Express.js je rychlý, flexibilní, robustní a minimalistický webový framework pro Node.js. Poskytuje funkce, které umožňují práci s HTTP požadavky, což zjednodušuje jejich zpracování a vývoj vlastní API. Vzhledem k tomu, že je Express.js pouze tenkou vrstvou nad Node.js, tak zásadně nezpomaluje jeho výkon, což je jeden z důvodů proč je tento framework součástí řady dalších populárních frameworků. [24]

6.4 EJS

Embedded JavaScript templates zkr. EJS je jednoduchý šablonovací jazyk, který umožňuje generovat HTML elementy pomocí JavaScriptu. Web tento jazyk používá na straně serveru, kde se pomocí EJS generuje HTML stránka, kterou následně webový prohlížeč načítá a zobrazuje. EJS umožňuje vytvářet šablony, kde lze pomocí speciálních značek vkládat dodatečný obsah, který je součástí finální HTML stránky ještě před jejím odesláním prohlížeči. Tímto způsobem je zajištěno efektivní znovupoužití stránek, jelikož se obvykle mění pouze malá část jejich celkového obsahu (např. jednotlivé stránky v rámci seznamu adres). Vzhledem k tomu, že jsou stránky z pohledu klienta statické a nevyžadují lokální použití JavaScriptu, tak jsou zároveň funkční i při jeho zakázání v prohlížeči. [25]

6.5 GitHub

GitHub je webová platforma pro ukládání a spravování kódu. Umožňuje sledovat změny kódu v čase a případně se vrátit k předchozím verzím software. Pro vytváření jednotlivých verzí software používá nástroj Git, což je open-source verzovací systém.

Při vývoji této práce je GitHub využíván převážně pro sledování změn při inkrementálním vývoji všech nástrojů řešení. Seznam změn je možné najít v historii commitů, kde každý jeden commit obsahuje krátký popis toho, k jakým změnám v daném commitu došlo. Součástí projektového repositáře je pak také značkovací soubor README.md, který obsahuje seznam funkcí, které by měly být implementovány s tím, že u každé je také grafické označení aktuálního stavu vývoje dané funkce. [26]

Kapitola 7

Závěr

Cílem této práce bylo zanalyzovat existující nástroje zabývající se vytvářením a udržováním databáze Bitcoinových adres a navrhnout vlastní prostředí, které vyřeší určité problémy spojené se stávajícími nástroji a případně zavede některé nové funkce, které u těchto nástrojů dosud chyběly.

Stanovené cíle byly v diplomové práci splněny. Mezi nejdůležitější funkce, které se o to zasadily je nutné uvést implementaci API, která mezi vyhledanými nástroji prakticky neexistuje. Většina nástrojů API nemá a ty, které ji mají nejsou v době psaní práce již funkční. Jediná BitcoinAbuse obsahuje funkční API, která však neumožňuje stahování celé databáze, ale pouze záznamů z posledních 30 dní.

Další funkcí je validace Bitcoinových adres, která u zkoumaných nástrojů opět velmi často chybí. Nástroje sice umožňují nahlášení kryptoměnových adres, nicméně nekontrolují, zda se skutečně jedná o validní Bitcoinové adresy. To pak vede k tomu, že weby sice uvádí, že je nahlášená adresa Bitcoinová, nicméně to ne vždy platí. Mimo jiné je to jedním z důvodů, proč téměř všechny obsahují nahlášení adres i jiných kryptoměn, i přestože se profilují jako čistě Bitcoinové. Právě k této kontrole je v implementovaném řešení využíváno seznamu Bitcoinových adres z blockchainu, jelikož adresy z blockchainu mají validitu zajištěnu.

Dále je určitě nutné zmínit podporu více kryptoměn. Mohlo by se zdát nesmyslné toto zmiňovat jako výhodu implementovaného řešení po tom, co bylo v odstavci výše kritizováno to, že některé řešení obsahují nahlášení více kryptoměn. Rozdílem je však to, že na těchto řešeních obvykle chybí jakákoliv informace o tom k jaké kryptoměně daná adresa patří a není možné také vyhledávat pouze adresy vybrané kryptoměny. Právě to však tato práce řeší, jelikož u každé adresy je její příslušnost k některé z podporovaných kryptoměn kontrolována a na webu i v API je možné procházet pouze adresy spojené s vybranou kryptoměnou.

Další užitečnou funkcí, která byla implementována je zobrazení celkového počtu nahlášených adres, který je navíc ještě rozlišený dle zdrojů a jednotlivých kryptoměn na daných zdrojích. Některé zdroje vlastní statistiky neuvádějí a je tak téměř nemožné spolehlivě určit, kolik adres obsahuje.

Tento nedostatek se mimochodem promítl i do statistik uvedených v této práci, kde v tabulce 2.1 nebylo u některých zdrojů možné určit celkový počet jejich adres.

Dále je možné uvést spolehlivý seznam adres. Některé zdroje neumožňují v seznamu přepínat na libovolné stránky, seznam obsahuje duplikátní adresy či tyto adresy nejsou seřazeny dle data poslední aktualizace. Všechny tyto nedostatky jsou v implementovaném prostředí vyřešeny.

Poslední funkcí, kterou lze vypíchnout je automatizovaný běh crawleru. I když tato funkce není příliš složitá na implementaci, tak velkým způsobem zjednodušuje použití crawleru. Ten tak může na zařízení běžet téměř bez zásahu uživatele a neustále poskytovat aktuální informace z vybraných zdrojů. Jediná zásadnější úprava je potřeba v případě potřeby přidání dalšího zdroje dat, úpravy některého ze stávajících zdrojů nebo pokud zestárne user-agent používaný při crawlování do fáze, kdy již nebude možné jej úspěšně použít.

I přes všechny implementované funkce je u této práce stále prostor pro možné rozšíření či vylepšení. Vyvíjené řešení například pracuje s omezením přístupu k databázi jednotlivým uživatelským rolím. U uživatelů sice lze uživatelskou roli upravit ve správě uživatelů, u dat však toto nastavení lze provést pouze v databázi. Na webu tak toto nastavení chybí. Uživatelská role tedy de facto ovlivňuje převážně limity použití API.

Dané řešení by mohlo také podporovat ještě více kryptoměn, jelikož jich dnes existuje již více než několik tisíc.

Adresy by poté mohly být získávány i z jiných zdrojů, které by nebyly zaměřeny pouze na nahlášení podvodů spojených s kryptoměnami. Muselo by však dojít k výrazné změně databáze, která s tímto aktuálně nepočítá a pravděpodobně by musely být porušeny pravidla `robots.txt`, které mohou být u některých stránek velmi omezující. Použití API je pak u některých webů téměř zbytečné, jelikož např. twitter poskytuje v bezplatné verzi pouze 1 500 požadavků za měsíc, což pro potřeby aktuálního řešení není v žádném případě dostačující.

Dále je pak prostor pro různé menší kosmetické vylepšení. Konzolová aplikace crawleru například obsahuje pouze základní informace o aktuálním stavu programu. Tyto informace by mohly být zobrazovány častěji a podrobněji, aby měl uživatel neustále přehled, o tom, co crawler právě zpracovává. Webová aplikace by pak mohla být lépe optimalizována pro mobilní zařízení. Aktuální implementace sice obsahuje prvky responzivního designu, avšak rozložení prvků na webu či jejich design nebyl jedním z primárních cílů řešení, takže je možné je ještě vylepšit.

Literatura

1. BADAWI, Emad; JOURDAN, Guy-Vincent; BOCHMANN, Gregor; ONUT, Iosif-Viorel. An Automatic Detection and Analysis of the Bitcoin Generator Scam. In: *2020 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)*. 2020, s. 407–416. Dostupné z DOI: 10.1109/EuroSPW51379.2020.00061.
2. BAMBUCH, Vladislav. *Platform for Cryptocurrency Address Collection*. Excel, 2020.
3. MACHARIA, Caroline Wanjira. *Maintaining a bitcoin address repository through focused web crawling*. 2017. Dis. pr. Strathmore University.
4. SANDERS, Dakota Blake. Oops, Did I Do That? Uncovering Apparent Mishandling, Tax Fraud, and Illegal Contributions via a Systematic Study of Cryptocurrency Donations to US Political Campaigns. 2020.
5. *VoteSmart* [online]. [cit. 2022-10-13]. Dostupné z: <https://justfacts.votesmart.org/>.
6. BARTOLETTI, Massimo; PES, Barbara; SERUSI, Sergio. Data Mining for Detecting Bitcoin Ponzi Schemes. In: *2018 Crypto Valley Conference on Blockchain Technology (CVCBT)*. 2018, s. 75–84. Dostupné z DOI: 10.1109/CVCBT.2018.00014.
7. GARBA, Abba; GUAN, Zhi; LI, Anran; CHEN, Zhong. Analysis of Man-In-The-Middle of Attack on Bitcoin Address. In: *ICETE (2)*. 2018, s. 554–561.
8. *LoyceV* [online]. [cit. 2023-02-24]. Dostupné z: <http://alladdresses.loyce.club/>.
9. *BlockChair* [online]. [cit. 2023-02-24]. Dostupné z: <http://blockchair.com/dumps>.
10. *BitcoinAbuse* [online]. [cit. 2022-10-19]. Dostupné z: <https://www.bitcoinabuse.com/>.
11. *CheckBitcoinAddress* [online]. [cit. 2023-02-24]. Dostupné z: <https://checkbitcoinaddress.com/abuse-reports-to-bitcoin-address>.
12. *BitcoinAIS* [online]. [cit. 2023-02-24]. Dostupné z: <https://bitcoinais.com>.
13. *BitcoinGeneratorScam* [online]. [cit. 2023-02-24]. Dostupné z: <http://ssrg.site.uottawa.ca/bgsieeesb2020/#urls>.
14. *CryptoBlacklist* [online]. [cit. 2023-02-24]. Dostupné z: <https://www.cryptoblacklist.io>.

15. *CryptoScamDB* [online]. [cit. 2022-10-19]. Dostupné z: <https://cryptoscamdb.org/>.
16. *SeeKoin* [online]. [cit. 2023-02-24]. Dostupné z: <https://www.seekoin.com/address.php>.
17. *Cryptscam* [online]. [cit. 2023-02-24]. Dostupné z: <https://cryptscam.com>.
18. *BitcoinWhosWho* [online]. [cit. 2023-02-24]. Dostupné z: <https://www.bitcoinwhoswho.com/>.
19. *How Much Bitcoin Is Lost Forever?* [online]. [cit. 2023-04-04]. Dostupné z: <https://www.hedgewithcrypto.com/how-much-bitcoin-is-lost/>.
20. *PostgreSQL* [online]. [cit. 2023-03-07]. Dostupné z: <https://www.postgresql.org/>.
21. *PostgreSQL vs MySQL: What is the Difference between them?* [online]. [cit. 2023-03-07]. Dostupné z: <https://blog.devart.com/postgresql-vs-mysql.html>.
22. *Node.js* [online]. [cit. 2023-03-07]. Dostupné z: <https://nodejs.org/>.
23. *What is the Node.js (Node) runtime environment? – TechTarget Definition* [online]. [cit. 2023-03-07]. Dostupné z: <https://www.techtarget.com/whatis/definition/Nodejs>.
24. *Express.js – Node.js web application framework* [online]. [cit. 2023-03-07]. Dostupné z: <https://expressjs.com/>.
25. *EJS – Embedded JavaScript templates* [online]. [cit. 2023-03-07]. Dostupné z: <https://ejs.co/>.
26. *What Is GitHub? A Beginner's Introduction to GitHub* [online]. [cit. 2023-03-07]. Dostupné z: <https://kinsta.com/knowledgebase/what-is-github/>.

Příloha A

Návod k použití

A.1 Příprava nástrojů

Pro zprovoznění nástrojů je nejprve potřeba nainstalovat vybrané programy 3. stran, které jsou vyžadovány danými nástroji a bez nichž není možné tyto nástroje spustit. V následujících 2 sekcích (A.1.1 a A.1.2) je postup instalace a přípravy rozepsán do bodů, jejichž pořadí je třeba se držet.

Dále je také třeba zmínit, že je nutné tyto programy instalovat ve verzi pro operační systém Windows, jelikož řešení bylo vyvíjeno a průběžně testováno v systému Windows 10 a později také ve Windows 11. Funkčnost v starších verzích Windows je možná, nicméně není zaručena. Pro další operační systémy pak dané řešení nebylo testováno.

A.1.1 Crawler

1. Stáhnout a nainstalovat PostgreSQL 15.2 – během instalace je nutné nastavit heslo: **postgres**
2. Stáhnout a nainstalovat Python 3.11 – během instalace je nutné vybrat možnost: přidat **python.exe** do PATH
3. V „Nastavení / Aplikace / Rozšířené nastavení aplikací / Aliasy pro spouštění aplikací“ vypnout **python.exe**
4. Restartovat počítač
5. Jít do projektového adresáře **db_builder**
6. Přejmenovat soubor **example_db.json** na **db.json**
7. Změnit heslo připojení k databázi v **db.json**
8. Přejmenovat soubor **example_setup.json** na **setup.json**

9. Změnit hesla uživatelů databáze v `setup.json`
10. Otevřít příkazový řádek
11. V příkazovém řádku nastavit aktuální pracovní adresář na projektový adresář `db_builder`
12. Nainstalovat balíčky knihoven pomocí příkazu: `pip install -U -r requirements.txt`

A.1.2 Web

1. Stáhnout a nainstalovat Node.js 18.15 LTS (verze s dlouhodobou podporou)
2. Restartovat počítač
3. Jít do projektového adresáře `client`
4. Přejmenovat soubor `example_db.json` na `db.json`
5. Změnit heslo připojení k databázi v `db.json`
6. Otevřít příkazový řádek
7. V příkazovém řádku nastavit aktuální pracovní adresář na projektový adresář `client`
8. Nainstalovat balíčky knihoven pomocí příkazu:
`npm i -g npm-check-updates && ncu -u && npm i`

A.2 Spuštění nástrojů

Jakmile jsou všechny potřebné programy nainstalovány a nástroje připraveny, je možné přejít k jejich spuštění. Nejprve je nutné spustit crawler, jelikož se při jeho prvním spuštění vytvoří databáze a nahrají se výchozí data. Chvilí poté je pak možné spustit web.

Vzhledem k tomu, že crawler nejprve začíná stahováním seznamu Bitcoinových adres z blockchainu, které trvá minimálně prvních přibližně 50-70 minut (dle specifikací daného zařízení), tak během této doby není možné očekávat jakákoliv data, které by bylo možné na webu zobrazit. Je to dáno tím, že je daná databázová tabulka během této doby uzamčená pro rychlejší zápis a tedy než je tento proces dokončen, tak se jeví jako by byla prázdná. Po dokončení tohoto procesu by pak již mělo být na webu možné zobrazit první získaná data, což je indikátor toho, že by měla být aplikace funkční. Pokud se u některých zdrojů zobrazují v příkazovém řádku crawleru chyby, či na webu data daného zdroje chybí, pak to znamená, že daný zdroj buď ještě nebyl crawlován či je aktuálně nebo dokonce trvale nedostupný. Dostupnost externích zdrojů není možné zaručit a tedy je vhodné aplikaci spustit co nejdříve, aby bylo těchto komplikací co nejméně.

Stejně jako u přípravy, tak i u spuštění jednotlivých nástrojů je uveden odpovídající bodový postup.

A.2.1 Crawler

1. Otevřít příkazový řádek (jako administrátor)
2. V příkazovém řádku nastavit aktuální pracovní adresář na projektový adresář `db_builder`
3. Spustit crawler pomocí příkazu: `python main.py`
4. V případě, že se zobrazí okno Řízení uživatelských účtů, tak vybrat možnost „Ano“

A.2.2 Web

1. Otevřít příkazový řádek
2. V příkazovém řádku nastavit aktuální pracovní adresář na projektový adresář `client`
3. Spustit web pomocí příkazu: `node main.js`

Web je následně možné navštívit ve webovém prohlížeči na adrese: „`http://localhost:3000`“.