Česká zemědělská univerzita v Praze

Provozně ekonomická fakulta

Katedra Informačního Inženýrství

Bakalářská práce

Decentralizovaná aplikace na blockchainové databázi

Rostislav Šlajs

ČESKÁ ZEMĚDĚLSKÁ UNIVERZITA V PRAZE

Provozně ekonomická fakulta

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Rostislav Šlajs

Informatika

Název práce

Decentralizovaná aplikace na blockchainové databázi

Název anglicky

Decentralized application based on blockchain database

Cíle práce

Práce je zaměřena na problematiku využití blockchain transakcí k realizaci systému pro uchování perzistentních dat. Cílem práce je decentralizovanou navrhnout a implementovat aplikaci na platformě Ethereum, která bude sloužit jako nástroj pro elektronické hlasování.

Metodika

Práce sestává se ze dvou částí, teoretické a praktické.

Teoretická východiska pro zpracování praktické části, představující první část práce, budou popsána na základě syntézy poznatků získaných studiem odborné literatury a dalších odborných a technických zdrojů. Součástí je průzkum možností využití blockchain technologií pro potřeby decentralizovaného ukládání dat.

Praktická část práce spočívá v analýze, návrhu a implementaci decentralizované aplikace pro provádění elektronického hlasování. Během analýzy a návrhu bude využito standardních metod softwarového inženýrství, implementace bude provedena s využitím jazyka JavaScript, Solidity/Truffle Suite Frameworku a dalších souvisejících technologií.

Aplikace bude otestována, budou shrnuty poznatky získané během jejího vývoje a navrženy případné další možnosti jejího využití.

Doporučený rozsah práce

35-40 stran

Klíčová slova

Blockchain, Ethereum, Bitcoin, smart contract, Solidity, Ethereum Client, Network, Geth, API Managment

---

Doporučené zdroje informací

Blockchain Technology: Introduction to Blockchain Technology and its impact on Business Ecosystem, Stephen Fleming, ISBN 1537841548, 9781537841540

Blockchain: Ultimate Guide to Understanding Blockchain, Bitcoin, Cryptocurrencies, Smart Contracts and the Future of Money, Mark Gates Pages, ISBN-10 : 1547090685 ISBN-13 : 9781547090686

Ethereum: Ethereum Homestead Documentation [online]. [cit. 2019-01-05]. Dostupné z: http://www.ethdocs.org/en/latest/

Solidity [online]. [cit. 2019-01-05]. Dostupné z: https://solidity.readthedocs.io/en/v0.5.3/

Předběžný termín obhajoby

2018/19 LS – PEF

Vedoucí práce

Ing. Jiří Brožek, Ph.D.

Garantující pracoviště

Katedra informačního inženýrství

Elektronicky schváleno dne 11. 3. 2019

Ing. Martin Pelikán, Ph.D.

Vedoucí katedry

Elektronicky schváleno dne 11. 3. 2019

Ing. Martin Pelikán, Ph.D.

Děkan

V Praze dne 14. 03. 2019

Čestné prohlášení

Prohlašuji, že svou bakalářskou práci "Decentralizovaná aplikace na blockchainové databázi" jsem vypracoval(a) samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou citovány v práci a uvedeny v seznamu použitých zdrojů na konci práce. Jako autor(ka) uvedené bakalářské práce dále prohlašuji, že jsem v souvislosti s jejím vytvořením neporušil autorská práva třetích osob.

V Praze dne 15. 3. 2019 _____

Poděkování

Rád(a) bych touto cestou poděkoval(a) Ing. Jiřímu Brožkovi, Ph.D z katedry informačního inženýrství, za odborné vedení mé bakalářské práce, cenné rady a čas, který mi věnoval při konzultacích.

Decentralizovaná aplikace na blockchainové databázi

Abstrakt

Tato bakalářská práce se zabývá koncepcí blockchain řešení, jejich technologiemi a jejich vývojem. Shrnuje poznatky o blockchainu a ukazuje výsledky reálného použití blockchainu, potažmo Smart kontraktů atd. V teoretické části se čtenář seznámí se základy blockchainu, jeho bohatou historií spojenou s Bitcoinem, fungováním a architekturou blockchainu společně s těžením v rámci blockchainu. Druhá část se týká Etherea jako platformy pro psaní chytrých kontraktů a celkového popisu fungovaní a vlastností Ethera. V poslední části teoretické práce se čtenář setká s programovacím jazykem Solidity. V této časti je popis základních funkcí, pro psaní chytrých kontraktů. V praktické části je popsán vývoj části webové aplikace, její architektura, use casy a specifikace problémové oblasti společně s předpoklady pro vývoj. Výsledkem je otestované demo decentralizované aplikace, schopné komunikovat s blockchain sítí. Aplikace je otestována a jej vytvořen testovací kontrakt, na který je možné hlasovat.

Klíčová slova: Blockchain, Ethereum, Bitcoin, DAPP, decentralizovaná distribuovaná databáze, peer to peer, chytrý kontrakt

Decentralized application based on blockchain database

Abstract

This bachelor thesis deals with the concept of blockchain solutions, their technologies, and development. It summarizes the knowledge of blockchain and shows the results of real use of blockchain, Smart contracts etc. In the theoretical part, the reader gets introduced into basics of blockchain, its rich history associated with Bitcoin, the functioning and architecture of blockchain together with the mining of blockchain. The second part deals with Ethereum, which is a platform for writing smart contracts and the overall description of Ethera's functionality and capabilities. The last part of the theoretical work is about programming language called Solidity. In this section, there is a description of the basic functions for writing smart contracts. The practical part describes the development of a part of the web application, its architecture, use cases, and specification of area together with development prerequisites. Result of this bachelor thesis is a tested demo of decentralized application capable of communicating with the blockchain network. The application is tested, and a test contract is created for it to vote on.

Keywords: Blockchain, Ethereum, Bitcoin, DAPP, decentralized distributed database, peer to peer, smart contract

Obsah

Seznam použitých obrázků

1 Úvod

Blockchain je jedno z mnoha „buzzwords“ dnešního světa. Většina společností se s ním snaží spojovat a investovat do něj peníze, jinak by byli, jak se říká „out“. Slovo „buzzword“ bychom si mohl popsat jako něco, co je aktuálně populární a o blockchainu se mluví již od vzniku Bitcoin. Jeho popularita stále narůstá a svojí unikátností by mohl pomoci oborům jako automobilový nebo finanční. Vlastnosti jako neměnnost nebo anonymita toto potvrzují.

Blockchain dovolil lidem obchodovat a vyměňovat si mezi sebou cokoliv digitálního bez prostředníka, banky, úřadu nebo správce na které jsme dnes zvyklí. Avšak i bez těchto prostředníku vše probíhá s maximální bezpečností. Toto je možné díky tomu, že na ověřování se podílejí všichni uživatelé této sítě. Tato činnost se nazývá těžení a není k němu potřeba žádná speciální technika, stačí osobní počítač. Blockchain lze přirovnat k účetní knize nebo excelovému listu, do kterého jsou zapisována data, například v případě Etherea každých patnáct minut. Toto se děje po vyřešení jakéhosi matematického přikladu, který svou obtížností stanovuje čas řešení. Dosáhnout výsledku může jak jedinec, tak i sdílená sít více počítačů. Motivací k řešení úloh je peněžní odměna ve formě kryptoměny. Odměna je rozdělena mezi jednoho či více účastníků a je zaslána na adresu jejich krypto peněženek. Zapisování dat do blockchainu může být prováděno díky Smart contracts („Chytrým kontraktům“), což je kus kódu v aplikaci, který je schopen komunikovat s blockchain sítí, získávat aktuální blok, komunikovat zpět kontrakt potřebný zapsat do distribuované databáze. (Crosby et al. 2016)

Cílem práce je popsat fungování blockchain technologie, nahlédnou do její historie, proč a jak vznikla. Dalším milníkem pro blockchain byl vznik Etherea, který umožnuje provádění chytrých kontraktů, i této platformě a kryptomněně je věnována v této bakalářské práci samostatná kapitola. V praktické části této práce je představen návrh hlasovací aplikace, součástí návrhu je i demo popisující a zobrazující základní funkčnost především chytrých kontraktů s webovou aplikací.

2 Cíl práce a metodika

2.1 Cíle práce

Cíl bakalářské práce je navrhnout software, který umožní hlasování s ukládáním výsledků do decentralizované databáze. Takovéto demo je z hlavní části zaměřeno na platformu Ethereum, možnosti dalšího rozvoje blockchainové aplikace a její schopnosti. Nástavbu dalších funkcí a jejich přínos a využitelnost pro dnešní společnost.

2.2 Metodika

Bakalářská práce je rozdělena na dvě části, první teoretická část zaznamenává popis technologie blockchain, jeho vznik fungování, části a funkce. Další teoretická část je popis Etherea, jako platformy pro budování decentralizovaných aplikací. V neposlední řadě bude tato práce popisovat teoretické části Solidity, jakož to programovací jazyk pro psaní chytrých kontraktů.

V praktické části bakalářské práce je navržena webová aplikace, zapisující výsledky hlasování do blockchainu. Je navrženo její webové rozhraní, persony atd. Dále bude navrhnuta její architektura, která je rozdělena do tří hlavních oblastí. Ve druhé časti praktické části bakalářské práce jsou zobrazeny stěžejní části kódu pro zprovoznění dema aplikace, funkčnost tohoto kódu bude popsána a vysvětlena. Výsledná aplikace bude otestována a výsledky z jejího testování budou prezentovány a ohodnoceny.

3 Teoretická východiska

3.1 Blockchain

Zjednodušeně můžeme blockchain popsat jako časově označené záznamy neměnných rekordů, které jsou na počítačové síti, kterou neovládá žádná entita. Každý blok obsahuje vepsaná data společně s odkazem na předešlý blok. Bloky jsou na sebe vázány použitím kryptografických principů, které tvoří „chain“. Dohromady toto tvoří blockchain, decentralizovanou sít, jejíž základními kameny bychom mohli popsat jako decentralizovanost, transparentnost a neměnnost. (Blockgeeks.com, n.d.)

3.1.1 Historie

Vznik blockchain se datuje k roku 2008 jako hlavní komponent k Bitcoinu, k podpoře transakcí této digitální měny. (Stephen, 2017) Na blockchain bylo od této doby pohlíženo jako na účetní knihu („public ledger“) všech transakcí. Blockchain v té době znamenal doslova řetěz bloků informací, které registrovaly Bitcoin transakce. Algoritmus a výpočetní infrastruktura vytváření, doplňování a používání bloků je považována jako blockchain technologie. Samozřejmě se k ní váže striktní set pravidel, která říkají, jak ověřit validitu bloku a zajistit to, že blok nebude upraven nebo smazán. (Zhao, Fan and Yan, 2016)

I když se vznik blockchain váže na Bitcoin, jeho aplikace se dnes již dostala mnohem dále než jen ke kryptoměnám. Existuje mnoho názorů, které říkají, že blockchain by mohl způsobit revoluci v mnoha oborech, jako například finance, účetnictví, právo atd.Toto vede ke třem generacím blockchainů, jménem Blockchain 1.0 pro digitální měny, Blockchain 2.0 pro digitální finance, and Blockchain 3.0 pro digitální společnost. (Zhao et al. 2016)

3.1.2 Fungování blockchainu

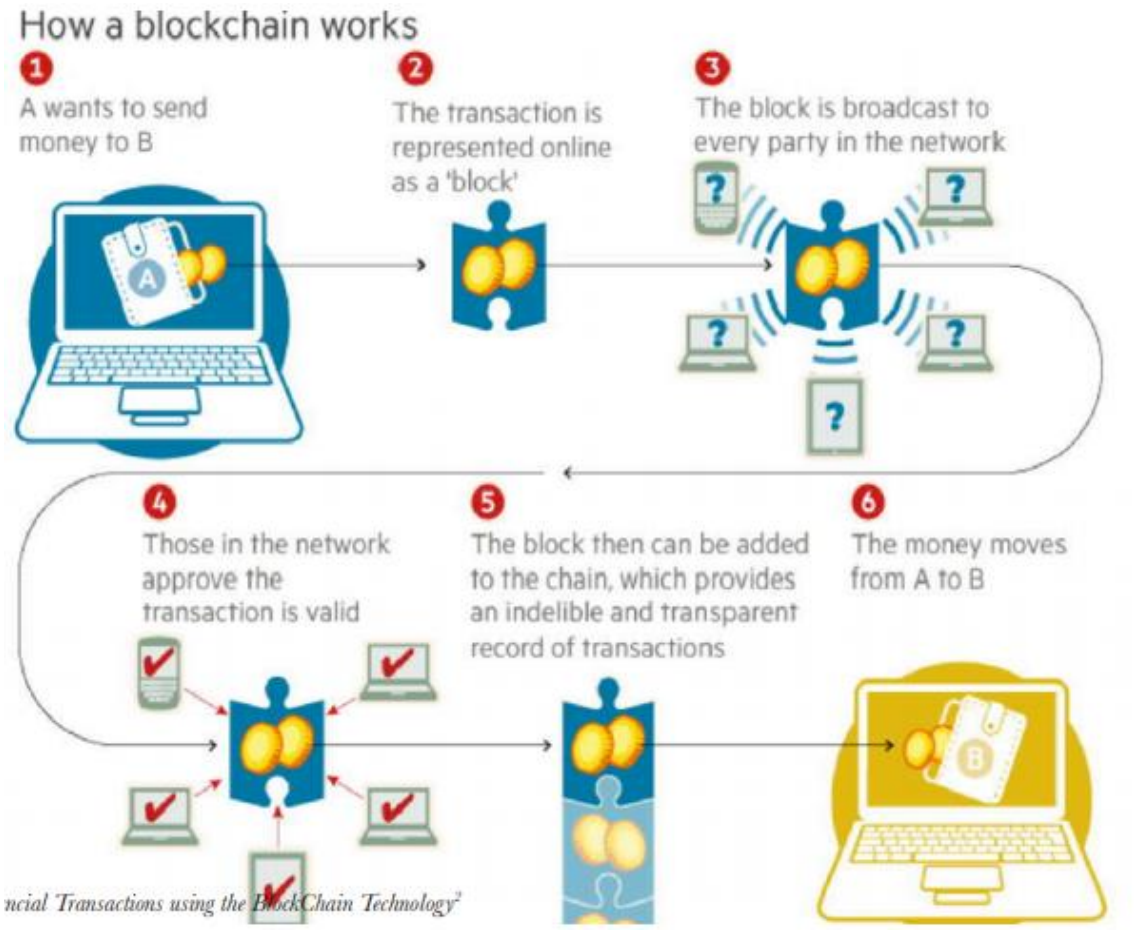
Vysvětlit, jak funguje blockchain lze na fungování Bitcoinu, jelikož blockchain je s Bitcoinem historicky spojen. Avšak technologie blockchainu je aplikovatelná na jakékoliv transakce digitálních aktiv online. (Crosby et al. 2016)

Bitcoin využívá důkaz („proof“) místo důvěry ve třetí stranu, která by v tomto případě sloužila jako důvěrník (Banky, PayPal, atd.). Každá transakce je odeslána na veřejný klíč příjemce a je digitálně podepsána privátním klíčem odesílatele. Za účelem nakládat s kryptoměnou musí majitel dokázat jeho vlastnictví privátního klíče. Subjekt přijímající digitální měnu, pak ověří digitální podpis, což znamená ověření vlastnictví korespondujícího privátního klíče, pomocí použití veřejného klíče na danou transakci. (En.bitcoin.it, 2019)

Každá transakce je pak rozeslána do každého uzlu v bitcoinové síti, verifikována a poté zapsána do veřejného bloku. Každá z proběhlých a zapsaných transakcí byla prověřena na validitu předtím, než byla nahraná do veřejného bloku. Uzly ověřující validitu, předtím, než nahrají jakoukoliv transakci, musí ověřit tyto dvě věci:

1. Odesílatel vlastní odesílanou hodnotu, ověření proběhne pomocí ověření digitálního podpisu na transakci. (Crosby et al. 2016)
2. Odesílatel má dostatečnou hodnotu kryptoměn na jeho učtu, ověřuje se kontrolou všech transakcí vedoucích proti účtu odesílatele, nebo jeho veřejného klíče. Toto zajišťuje dostatečný zůstatek na účtu před finalizací transakce. (Crosby et al. 2016)

Vyzvedá se otázka, jak udržovat pořádek mezi všemi transakcemi, které jsou odeslány všem uzlům blockchainové sítě. Transakce nepřicházejí do uzlů, v pořadí, ve kterém byly vytvořeny, a tím pádem je potřeba zajistit systém, který zamezí dvojitému utracení kryptoměn. Bitcoin vyřešil tento problém, mechanismem s názvem Blockchainová technologie. (En.bitcoin.it, 2019) Bitcoin systém řadí transakce umisťováním je do bloků a poté spojováním těchto bloků napříč Blockchainem. Transakce v jednom bloku se považují chronologicky a každý nový blok obsahuje “haš“ předešlého bloku viz Obrázek 1. (Crosby et al. 2016)



Obrázek 1 Finanční transakce provedená použitím Blockchainové technologie (Crosby et al. 2016)

3.1.3 Blok

Blok v Ethereum je kolekce relevantních informací, označovaných jako hlavičky bloků, společně s informacemi odpovídajícími tvořené transakce a množině ostatních informacích o předcházejících blocích, o které je známo že má rodiče, který se rovná současnému bloku rodiče rodiči (takovéto bloky se také nazývají ommers (Hallion, 2018)). (Wood, n.d.)

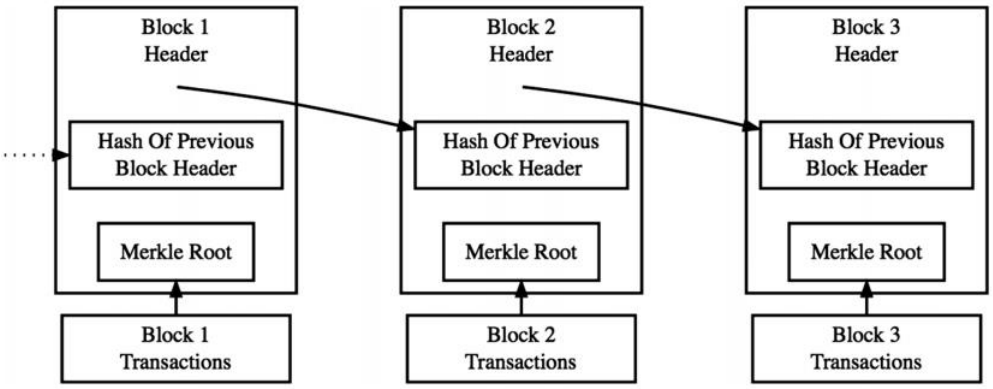
Hlavičky bloků obsahují několik dílků informací:

- parenHash: haš hlavičky rodičovského bloku v celém svém rozsahu, k hašování je použita SHA-3 kryptografická hašovací funkce.
- ommerHash: haš ommer dílu informací tohoto bloku, k hašování je použita SHA-3 kryptografická hašovací funkce.
- beneficiary: 160 bitová adresa, na kterou budou převedeny všechny poplatky z úspěšného minování, tohoto bloku.

- stateRoot: haš kořenového uzlu stavu pokusu, potom co jsou provedeny všechny transakce a dokončení aplikováno, k hašování je použita SHA-3 kryptografická hašovací funkce.
- transactionRoot: haš kořenového uzlu stavové struktury, osídlený s každou transakcí v transakčním listu v bloku, k hašování je použita SHA-3 kryptografická hašovací funkce.
- receiptsRoot: haš kořenového uzlu stavové struktury, osídlený dokladem o každé transkaci v transakčním listu v části bloku, k hašování je použita SHA-3 kryptografická hašovací funkce.
- logsBloom: filter složený z vázaných informací, obsahovaných v každé zapsané vstupu, z dokladu každé transakce v transakčním listu.
- difficulty: Skalární hodnota korespondující s obtížností výpočtu tohoto bloku. Toto lze vypočítat z obtížnosti předchozího bloku a časového označení (timestamp).
- number: Skalární hodnota rovna číslu předchozích bloků. Počáteční blok má hodnotu 0.
- gasLimit: Skalární hodnota rovna současnému limitu vynaloženého gas na blok.
- gasUsed: Skalární hodnota rovna součtu gas použitému v transakcích v bloku.
- timestamp: Skalární hodnota
- extraData: libovolná array bytů obsahující relevantní data pro tento blok. Toto musí být maximálně 32 bytů.
- mixHash: 256 bitový haš, který dokazuje společně s nonce, že byla provedená dostatečná kalkulace na tomto bloku.
- nonce: 64 bitový haš který dokazuje společně s mix-hash že byla provedená dostatečná kalkulace na tomto bloku.

Další dva komponenty v bloku jsou, list ommer blok hlaviček (který má stejný formát jako popsáno nahoře) a série transakcí. (Wood, n.d.)

Dohromady s předchozími hlavičkami bloků tvoří řetěz, každý blok také obsahuje „Merkle root“, nazývaný taky jako haš všech hašů. Tento haš slouží k zjednodušení celkové validity, jelikož díky němu nemusí být stahován celý blockchain, a obsahuje, jak napovídá jeho název haš všech předchozích hašů v každém bloku, viz obrázek 2. (de Kruijff and Weigand, 2017)



Obrázek 2 Zobrazení blockchain transakcí v blocích (de Kruijff and Weigand, 2017)

3.1.4 Hašový strom

Každý blok v bitcoin blockchainu obsahuje součet všech transakcí v bloku používajícího hašový strom. Hašový strom je datová struktura používaná pro efektivní shrnutí a ověření integrity velkých datových setů. Hašové stromy jsou binarní stromy obsahující kryptografické haše. Pojem strom je používán zejména k popsání větvovité datové struktury. (Antonopoulos, 2014)

Hašový strom je používán v Bitcoin k shrnutí všech transakcí v bloku, produkuje tím celkový otisk celého setu transakcí, poskytující velice efektivní proces k ověření, jestli byla transakce zahrnuta v bloku. Konstrukce Hašového stromu je rekurzivní, spojováním hašovaných párů uzlů, dokud nezbyde jeden poslední haš, tento haš se nazývá „kořen“. Algoritmus pro hašovaní v bitcoin hašovacím stomě je SHA256. Jelikož je hašovací strom binánárním stromem, potřebuje sudý počet listů uzlů. Pokud je počet transakcí lichý je poslední haš transakce duplikován k vytvoření sudého binárního stromu, takovýto strom se pak nazývá vyvážený strom. (Antonopoulos, 2014)

Zjednodušené platební ověřování („Simplified Payment Verification“ dále jen jako SPV) používá velice často hašové stromy. SPV uzly nemají všechny transakce a nestahují celé bloky, jenom jejich hlavičky. Pro ověření zdali je transakce přítomna v bloku, bez nutnosti stahovat všechny transakce v bloku, užívají hašovou cestu. (Antonopoulos, 2014) Toto zjednodušení dělá komunikaci mnohem rychlejší a méně náročnou díky menšímu objemu stažených dat z blockchainu.

3.1.5 Těžení

Těžení neboli „mining“ je nedílnou součástí procesu, ve kterém generace, transmise a validace transakce kryptoměny je provedena. (Krishnan, Saketh and Tej, 2015) Těžaři hrají dominantní roli v těžení. Těžaři zpracovávají transakce ověřováním vlastnictví měny ze zdroje do destinace. Každá transakce obsahuje haš předchozí provedené transakce, vygenerovaný vlastníkem, přes který se ověřuje autenticita aktuální transakce validací. Těžaři také zabraňují dvojitému utrácení tímto validačním procesem. Hlavním smyslem těžení je generovat a vypouštět mince na trh. Kdykoliv, kdy je transakce provedena a je validována, je sebrána těžařem a zařazena do bloku, který je momentálně těžen. Každý blok musí být vytěžen před jeho vysláním na blockchain. Těžení je řešení matematických problémů, puzzlů, které jsou náročné vyřešit. Jejich vyřešením je vygenerována výsledná hodnota, toto je jediný způsob, jak zapsat blok do blockchain a jako odměna se poskytuje těžaři malá hodnota dané kryptoměny. Toto vede k soupeření těžařů mezi sebou, který z nich dokáže rychleji vyřešit zadaný matematický problém. Takovýto mechanismus zároveň zachovává férovost v systému, jelikož nenadržuje nikomu. (Krishnan et.al. 2015)

Těžení se provádí skrz speciálně navrhnuté a sestrojené „těžební sestavy“. Historicky se tyto sestavy posunuly od CPU k ASIC designu. Periodický růst náročnosti matematických problémů, vedl k evoluci sestav. Náklady na pořízení a výkonnost těžební sestavy udává její profitabilitu, z toho důvodu je velice důležité klást důraz na design a jeho implementaci v těžení. (Krishnan et. al. 2015)

3.1.5.1 CPU

V počátečních dnech těžení, Cpu byly využívány k efektivnímu těžení s haš kurzem do 10MH/sec. Osobní počítač s těžebním softwarem byl tím pádem schopný pracovat jako těžební stanice. Nicméně kvůli již zmíněnému nárůstu obtížnosti jednotlivých matematických problémů, se CPU těžení stalo bezvýznamné. Jedním z nejpopulárnějších softwaru na těžení byl „cpuminer“. Tento program umožňoval jak sólo, tak i sdílené těžení. V podstatě program dostal každý navržený blok ze sítě, a snažil se uhodnout nonce hodnotu, která by vedla k validnímu bloku. V případě sdíleného těžení, síť je povinna rozdělit kredit mezi všechny těžaře, kteří se podíleli na těžení daného bloku. Daná částka je jim pak zaslána na adresu jejich krypto účtu. (Krishnan et.al. 2015)

3.1.5.2 GPU

S klesající sílou CPU sestav, přišli na svět CPU sestavy s grafickou kartou, obsahující grafické procesní jednotky (GPU‘s). (En.bitcoin.it, 2019) Grafické karty jsou využívány k řešení těžkých matematických kalkulací a komplexních polygonů v herním průmyslu. Tyto vlastnosti jim umožnují celkem dobře těžit kryptoměny. Jelikož jsou takovéto sestavy ponechány k těžení několik měsíců, jejich uživatelé se snaží agresivně upravovat jejich voltáž, k zmenšení nákladů na těžení. (Krishnan et. al. 2015)

V dnešní době je GPU těžení ve většině již neaktuální a nikdo je již neprovozuje. Díky stále narůstající náročnosti matematických problémů a neschopnosti soupeřit s architekturami jako ASIC v profitabilitě. (Krishnan et. al. 2015)

3.1.5.3 FPGA

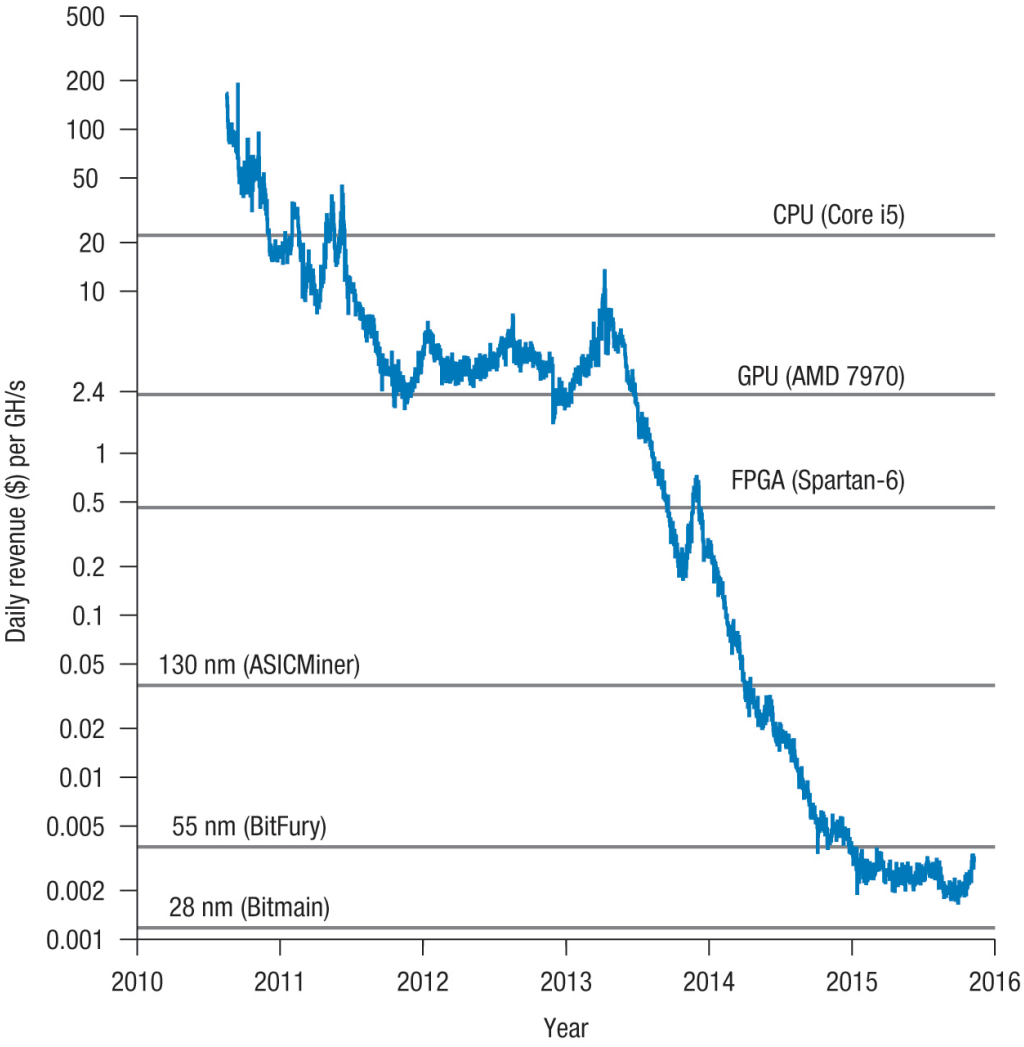
FGPA známé jako reprogramovatelné integrované obvody, které mohou být konfigurovány nebo designovány po výrobě. FGPA mohou obsahovat individuální logické bloky, tyto bloky jsou pospojovány, ale jejich spojení může být přeorganizováno. K počítání komplexních úkolů jsou v FGPA velké zásoby logických bran a RAM pro komplexní digitální počítání. Jednou z jejich hlavních výhod oproti GPU je, že spotřebovávají o jednu pětinu méně energie. Bohužel cena FGPA při produkci stoupá s vyšším počtem těchto tařízení, tím pádem jsou méně efektivnější než ASCI, jejich konkurence. (Krishnan et. al. 2015)

3.1.5.4 ASIC

ASIC jsou mikročipy vyrobeny pro jediný účel, avšak jejich použití je zaznamenáno v několika oblastech. (AnySilicon, 2018) Bitcoin ASIC jsou designovány specificky k těžení Bitcoinu, jsou dobré v počítání komplexních matematických úkolech, při maximální efektivnosti a rychlosti. Nejlepší ASCI na trhu dnes dokáží něco přes 1,000 Mhash (1 miliarda hašů) na jeden joul elektrické energie. Tyto vlastnosti dělají z ASCI nejlépe vydělávající privátní těžební sestavy dostupné momentálně na trhu. (Krishnan et. al. 2015)

3.1.5.5 Porovnání HW

Na grafu v Obrázku 3 je vidět výnos per GH/s, který vygenerovala bitcoin sít od roku 2010. Podle křivky v grafu je možné určit, kdy je pod denním výnosem a dle dané sestavy se jí již nevyplatí provozovat.



Obrázek 3 Výnos za GH/s v letech (Krishnan et. al. 2015)

3.2 Ethereum

Ethereum je projekt, který se pokouší o budování obecné technologie, technologie, na které všechny koncepty transakčních projektů mohou být vybudovány. Krom tohoto se snaží poskytnout koncovým developerům pevně integrovaný end-to-end systém pro budování softwaru. (Wood, n.d.) Ethereum dosahuje tohoto pomocí budovaní ultimátí abstraktní základové vrstvy: blockchain s již zabudovaným programovacím jazykem, umožňujícím komukoliv psát chytré kontrakty a decentralizované aplikace, ve kterých je možno si nastavit svoje vlastní pravidla vlastnictví, formáty transakcí a funkce stavu transakcí. (Buterin, 2014)

Hlavním rozdílem mezi Bitcoinem a Etherem je, použití Bitcoinu jako distribuované databáze pro finanční transakce, Ethereum je designováno pro použití jako distribuovaná výpočetní platforma pro aplikace. To znamená, že Bitcoin může být použit k placení kdekoliv, kde je akceptován, kryptoměna Ether, která byla vytvořena pro sít Ethereum, byla vymyšlena pro developery k placení výpočetní síly na síti, za jejich běžící decentralizované aplikace. Bitcoin a Ethereum mají obě své digitální měny, ale každá je využívána k jinému záměru. Hlavním smyslem Etheru není stát se alternativou k Bitcoin, ale povzbuzovat developery k vytváření decentralizovaných aplikací v Ethereu. (Gates, 2017)

Ethereum blockchain je masivní sít tísíců počítačů z celého světa. Ethereum bere dohromady všechny připojené počítače a poskytuje jejich výpočetní sílu developerům k vytváření aplikací. Díky tomu nemusejí developeři tvořit jejich vlastní blockchain a shánět pro něj výpočetní sílu. Ethereum má zároveň „Ethreum Virutal Machine“ kompilátor a „Solidity“ programovací jazyk. (Buterin, 2014) Solidity je používán k tvoření decentralizovaných aplikací nebo chytrých kontraktů, které potom kompiluje dohromady EVM, a jsou poslány na blockchain. (Gates, 2017)

3.2.1 Chytrý kontrakt

První zmínka o chytrých kontraktech se datuje k roku 1993, kde Nick Szabo popsal, jak mohou uživatelé vložit data nebo hodnotu, a získat určitý předmět zpět. Bitcoin se tak zapsal jako první kryptoměna podporující chytré kontrakty, ve smyslu posílaní měny mezi uživateli. Výhodou Etherea je jeho volnost při tvoření chytrých kontraktů, developeři mohou psát své vlastní, do svých vlastních programů a aplikací. (Hertig, n.d.) Chytré kontrakty jsou programy, které řídí chování účtů v Ethereu. (Kasireddy, 2017) Zrcadlí skutečné dohody a smlouvy reálného světa ve světě kybernetickém. (Swan, 2015)

Klíčový předpoklad pro kontrakt je reprezentace dohodnutého závazku mezi dvěma nebo více stranami, kde každá z těchto smluvních stran musí splnit povinnosti zavázané v kontraktu. Důležitou věcí je domluva podle zákona, ochráněna většinou nějakou statní institucí, tomuto se nicméně chytré kontrakty vyhýbají. Vymahatelnost je pak dosažena pomocí automatického vykonávání kódu, který je distribuován a kontrolován uzly sítě v decentralizované blockchainové síti. Chytré kontrakty dále umožnují kontrakty mezi

nedůvěryhodnými stranami, bez zprostředkovatelských poplatků, závislostí na třetích stranách a potřebě přímé interakce mezi stranami. (Swan, 2015)

3.2.2 Ethereum Virtual Machine

EVM je bezpečně orientovaný virtuální stroj, designovaný k povolování spouštění nedůvěryhodného kódu na globální síti počítačů. (Buterin, 2014)

transakce okud je nasazen kód kontraktu, nemůže již být změněn. Hašový strom kontraktu/účtu je pozměněn po jakékoliv úspěšné transakci, pokud výsledná hodnota je uložena pod novým klíčem nebo na existující klíč. (Buterin, 2014)

3.2.3 Hašovací Patricia strom

Patricia strom je datová struktura podobná hašovému stromu, může být také nazývana jako „trie“. Tato datová struktura užívá klíč jako cestu, takže uzly se stejnou předponou mohou sdílet jednu cestu. Tato struktura je rychlejší v hledaní běžných předpon, jednoduchá k implementaci a vyžaduje malou paměť. (Kim, 2018)

3.2.4 Poplatky

Každá transakce, která se objeví na Ethereum síti obsahuje poplatky, tyto poplatky jsou placeny v hodnotě zvané palivo („gas“). Palivo je jednotka měřící velikost poplatku potřebnou pro určenou kalkulaci. Cena paliva je množství Etherea, kterou je uživatel ochotný zaplatit za jednotku paliva. Každá jednotka paliva je meřena v „gwei“, „Wei“ je nejmenší jednotka Etherea, hodnota je 1^{018} Wei se rovna jednomu Etheru. Jeden gwei je 1 miliarda Wie.

S každou transakcí definuje odesílatel dvě hodnoty, limit paliva a cenu paliva. Výsledek těchto dvou hodnot je maximální počet Wei, kolik je schopný odesílatel zaplatit za transakci.

Pokud je cena transakce menší, než poskytnutý počet paliva je zbytek paliva vrácen uživateli zpět. V opačném případě, pokud není dostatek paliva na zpracování je transakce považována za nevalidní. Tento výsledek je zapsán do bloku, zobrazující pokus o zpracování transakce, jelikož však bylo již palivo vynaloženo na pokus o zpracovaní není co vrátit odesílateli zpět. (Kasireddy, 2017)

Poplatky jsou sbírány k motivaci těžařů, za jejich práci. Komu se podaří vytěžit blok je převedeno na jeho adresu sebraný poplatek, jako odměna. (Kasireddy, 2017)

3.2.5 Proof of Stake

Alternativa k „Proof of Work“ je „Proof of Stake“ (dále je jako PoS) protokol. Ten uděluje rozhodovací moc všem, kteří mají podíl v systému. Oproti POW, ve kterém se může kdokoliv stát těžařem, oproti tomu né každý se může připojit k PoS síti. Vlastnictví měny nebo vložená částka v síti dovoluje uzlům podílet se na ražení, validaci transakcí a tvorbě bloků. Žádná početní síla není potřeba k řešení matematických problému jako v PoW schématu. Nejsou zde žádné odměny ve formě tvorby peněz. Kontroloři sbírají poplatky od uživatelů a jsou z nich placeni jako obyčejný zprostředkovatelé. Jelikož kontroloři získávají jenom poplatky z transakcí, je vyloučeno že by tvořili prázdné bloky, naopak jejich cíle je zahrnou do bloku co nejvíce transakcí, aby maximalizovali svůj zisk. Jelikož nebudou vytvořeny žádné nové mince, v čisté formě PoS, tak peněžní zásoba musí být vydána a spravedlivě rozdána na počátku. (SEANG and TORRE, 2018)

Ethereum momentálně používá PoW protokol, ale plánuje přejít v budoucnu na PoS protokol, přesné datum ještě nebylo uvedeno. (Buterin, 2014) Zakladatel Etherea Biterin o tom informoval ve svém článku v roce 2014.

3.3 Solidity

Solidity je objektově orientovaný, vyšší programovací jazyk propsaní chytrých kontraktů. Vývoj Solidity byl ovlivněn C++, Pythonem a Javascryptem a designován pro „Ethereum Virtual Machine“. Solidity je statický typizované, podporuje dědění, knihovny a komplexní uživatelské typy mezi mnoha dalšími funkcemi. (Solidity.readthedocs.io, 2016)

Díky solidity lze tvořit kontrakty pro volení, aukce, sbírky atd. Kontrakty v Solidity jsou velice podobné třídám v objektově orientovaných jazycích. Každý kontrakt může obsahovat deklarace stavové proměnné („State Variable“), to jsou proměnné, které jsou permanentně uloženy v kontraktu. Funkce („Functions“), spustitelné časti kódu uvnitř kontraktu, mohou mít různé stavy viditelnosti k ostatním kontraktům a jsou prováděny interně nebo externě. Funkce akceptují parametry a vracejí proměnné k předávání parametrů a hodnot mezi nimi. Modifikátory funkcí („Function Modifiers“) k pozměnění, opravě sémantiky funkce v deklarativním směru. Události („Events“), jsou užitečné v propojení s EVM logovacím zařízením. V neposlední řadě také „Struct Types“ a „Enum Types“ k definování vlastních skupin proměnných a k definování omezených setů kontraktních hodnot. Kontrakty mají také tu vlastnost, že mohou dědit z jiných kontraktů. (Solidity.readthedocs.io, 2016)

4 Vlastní práce

Praktická část této práce je dedikována návrhu hlasovací decentralizované aplikace. Tento návrh se bude skládat od základní specifikace architektury, use casů, a částí kódu z implementace chytrého kontraktu. Pro práci chytrými kontrakty je použit programovací jazyk Solidity, front end aplikace je tvořen pomocí Node.js. Pro účel této práce je navrhnuto pouze demo později specifikované hlasovací aplikace. Toto demo znázorňuje podle autora nejdůležitější věci při vývoji decentralizované aplikace a zároveň poskytuje nový pohled na práci s daty v dnešním světě plném serverových aplikací.

4.1 Specifikace

Účel této práce je přiblížit práci s Etherea jako poskytovatele decentralizované sítě pro chytré kontrakty psané v Solidity programovacím jazyku.

Volební aplikace by měla sloužit k uchovávání výsledků, jakýchkoliv dotazů, průzkumů a hlasování, v blockchainu. Pro tento účel zapisování je zde Solidity, tento zápis dat do bloků plně nahrazuje jakoukoliv nutnost ukládání dat do databáze. Ethereum bylo zvoleno jako platforma pro blockchainu. Zapisování do bloků Etherea probíhá pomocí volání jednotlivých skriptů/funkci/kusů kódu. Kdokoliv si může vytvořit svoje vlastní hlasování, stejně jako každý hlasující musí zaplatit poplatek za zpracování tohoto požadavku do blockchainu.

Celá aplikace je napsána pomocí Node.js javascriptové nástavby, HTML a CSS se starají o front end. Pro komunikaci s blockchainem, který zde působí jako „back end“ je použito Solidity na smart kontrakty kompilované Trufflem, který je tu jako Framework pro testování blockchainu používajících EVM kompilátor. (Truffle Suite, 2019) Nosič informací o hlasování je URL každého vytvořeného hlasování. Jako prohlížeč je doporučen Mist.

4.2 Identifikace problémové oblasti

Implementace blockchainové technologie do webové aplikace pro záznam hlasování, může být v budoucnu velice žádanou věcí. A to díky téměř neměnnému zapisování záznamů do bloků, které jsou komukoliv otevřeny k nahlédnutí a kontrole. Tato aplikace je zasazena do webového prostředí, aby si zajistila co největší dosah ve virtuálním světě a zároveň je koncipována na bázi využívání zdarma.

4.3 Personifikace

Tato aplikace je zaměřena pro všechny obyvatele, nehledě na věku a technických znalostech. Ovládání musí být intuitivní, jednoduché a dostatečně přehledné pro jakéhokoliv člověka, který má možnost se setkat s touto aplikací.

Hlasovací aplikace je zaměřena na:

- Věková kategorie od 10 do 99
- Na uživatelé, co mají komunitu, se kterou mohou takto lehce interagovat
- Na širší publikum uživatelů dotazovaných takovýmito hlasovacími prostředky

4.3.1 Uživatelé – vzorové persony

4.3.1.1 Persona 1

Jan Novák

- Muž
- 36 let
- Zaměstnaný, ženatý
- Dělník ve stavební firmě
- Koníčky: sledování televize, „hoax news“, brouzdaní po internetu

Jan je velice komunikativní člověk, je spíše levicově nakloněný a má rád Rusko. Každý den, když přijde domů z práce, čte a píše články na jeho oblíbený novinový server.

Přínos hlasovací aplikace je pro něho v tom, že umožnuje jeho komunitě čtenářů zvolit si další téma na článek.

Přínos aplikace: průzkum názoru a mínění čtenářů na daný obsah, generace budoucího obsahu

4.3.1.2 Persona 2

Petr Novotný

- Muž
- 24 let
- Zaměstnaný, svobodný
- Streamer a youtuber
- Koníčky: hraní pc her, stříhání videí, venčení psa

Petr má rád hraní her, okolo této aktivity si vybudoval i svou kariéru. Má tisíce followerů na sociálních sítích a pořádá pro svoje fanoušky četné akce. Celý den tráví natáčením videí nebo streamováním a jednou týdně pořádá speciální event. Chce aby se jeho fanoušci co nejvíce zapojili, proto jim umožnuje volit témata těchto akcí.

Přínos aplikace: generace obsahu pro tvůrce, povědomí o stavu komunity, navazování spojení s komunitou

4.3.1.3 Persona 3

Marek Svoboda

- Muž
- 13 let
- Nezaměstnaný, svobodný
- Žák základní školy
- Koníčky: sledování videí na internetu, hraní pc her

Marek je ještě školák. Miluje svoje oblíbené youtubery a streamery a sleduje je každý den. Pokud se mu naskytne nějaká šance dát vědět svému idolu, co si myslí o tom, co zrovna

frčí nebo jakou hru by chtěl vidět na streamu příště, moc rád to udělá. Hlasuje v každém průzkumu, o kterém ví nebo mu kamarádi řeknou, že je dostupný.

Přínos aplikace: Navázání kontaktu s oblíbeným youtuberem/streamerem.

4.4 Use Case

4.4.1 Use Case – úvodní strana

Uživatel se otvírá webovou aplikaci, předpokládá zadání:

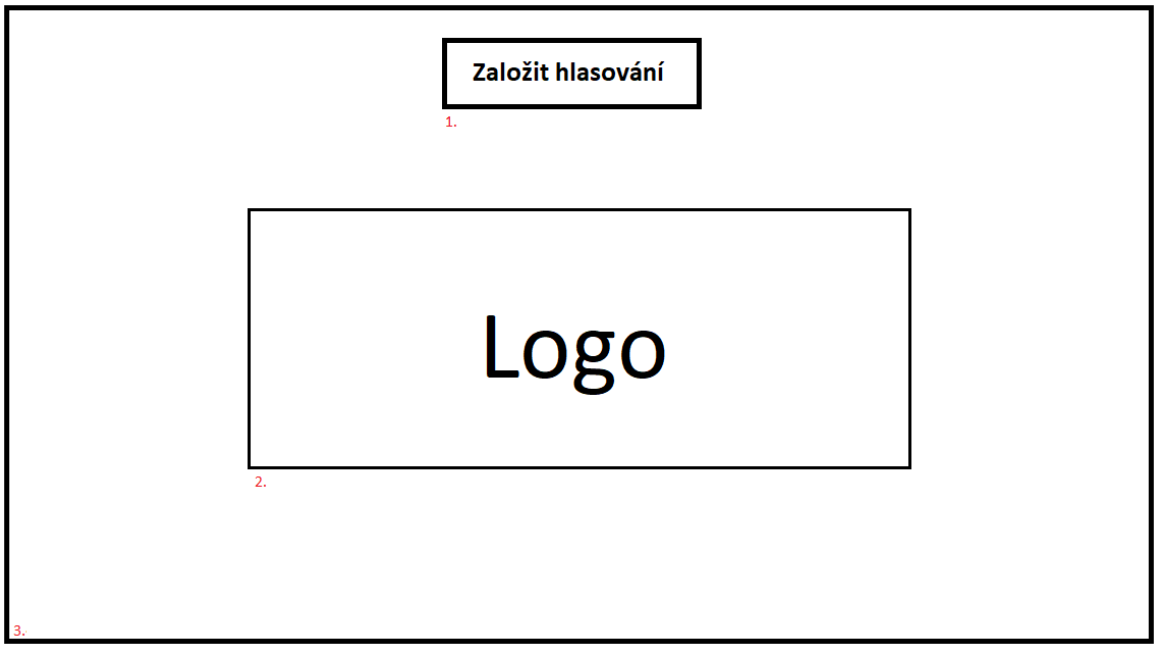
- Po kliknutí na tlačítko pro založení hlasování, očekává zobrazení stránky umožnující mu zadat potřebné informace a založit hlasování

4.4.2 Scénář – úvodní strana

Systém načte stránku.

- Systém zobrazí hlavní stránku s aktivním prvkem „Založit hlasování“
- Proběhne otevření odkazu, po kliknutí
- Zobrazí se stránka 4.4.4.1 Logický design – vytvoření hlasování

4.4.2.1 Logický design – úvodní strana



Obrázek 4 Úvodní strana, logický design

1. Tlačítko založení hlasování
2. Prostor pro logo
3. Volný prostor

4.4.3 Use Case – vytvoření hlasování

Uživatel očekává vytvoření hlasování:

- Otázka hlasování
- Odpovědní možnosti
- Uložení rozpracovaného formuláře
- Vytvoření odkazu na formulář

4.4.4 Scénář – vytvoření hlasování

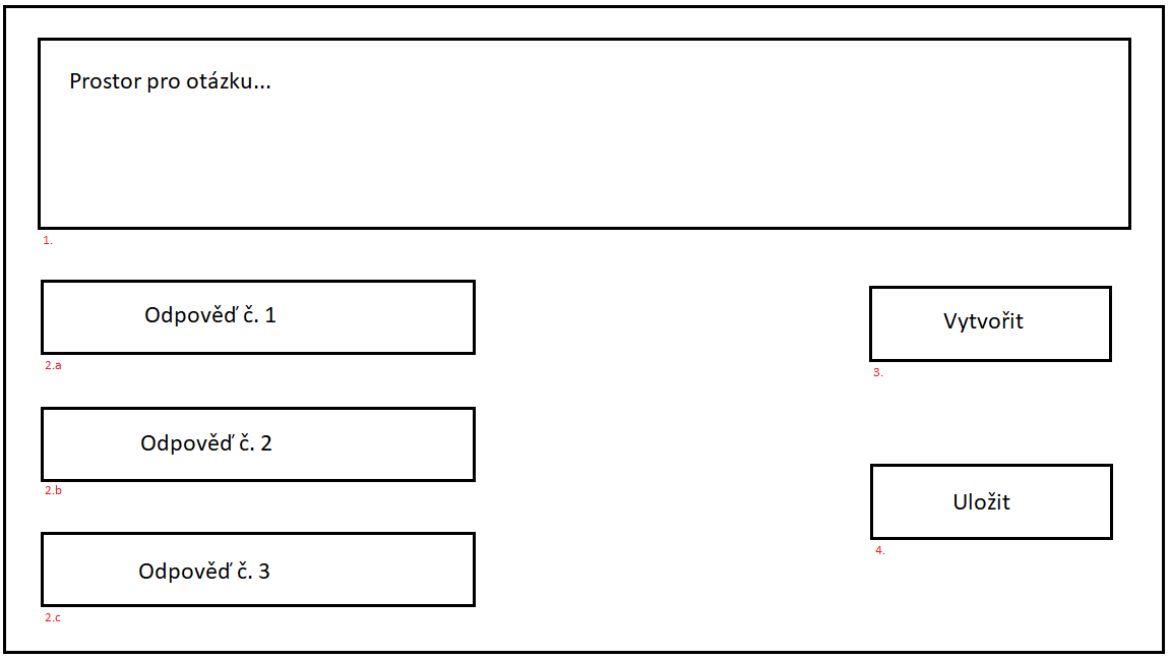
Stránka nabízí založení hlasování pomocí tlačítka – „Vytvořit hlasování“

- Systém načte stránku s prvky:
  - Otázka
  - Možnosti odpovědí 1 až 3 (po vytvoření 3 odpovědi se automaticky načte další řádek

- Tlačítko – „Vytvořit“
- Tlačítko – „Uložit“

- Po vložení těchto údajů může uživatel vytvořit hlasování, odkaz na toto hlasování je mu zobrazen v novém vyskakovacím okně s prvkem:
  - URL odkaz
- Odtud je možno vykopírovat odkaz na hlasování.

4.4.4.1 Logický design – vytvoření hlasování

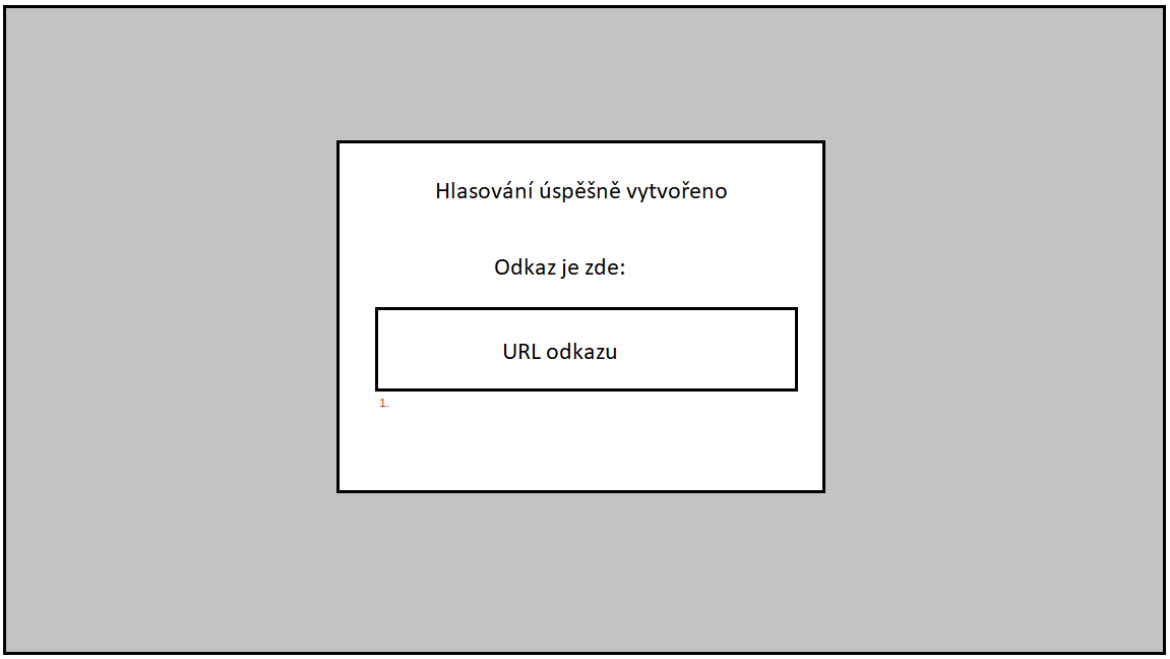


Obrázek 5 Vytvoření hlasování, logický design

Obrázek č.4 Vytvoření hlasování, logický design

1. Prostor pro text
2. Odpovědi
3. Tlačítko na vytvoření hlasování
4. Tlačítko na uložení rozpracovaného dotazníku

4.4.4.2 Logický design – URL odkazu hlasování



Obrázek 6 URL odkazu hlasování, logický design

1. Prostor pro vykopírování URL odkazu na sdílení hlasování

4.5 Architektura

Architekturu můžeme rozdělit do tří nejdůležitějších částí, těmi jsou:

- Smart kontrakty,
- Uživatelské Interface,
- Transakční Interface

(State of the ĐApps Blog, 2018)

4.5.1 Smart kontrakty

Základní věcí je psát chytré kontrakty co nejjednodušší, díky tomu jsou poplatky co nejmenší. Při psaní kontraktů bychom se měli držet jedné věci, a tou je rozdělovat business logiku a data model do rozdílných kontraktů. Pak je oba spojit hlavním kotraktem, který je hlavní chytrý kontrakt, který ovládá celou aplikaci. (State of the ĐApps Blog, 2018)

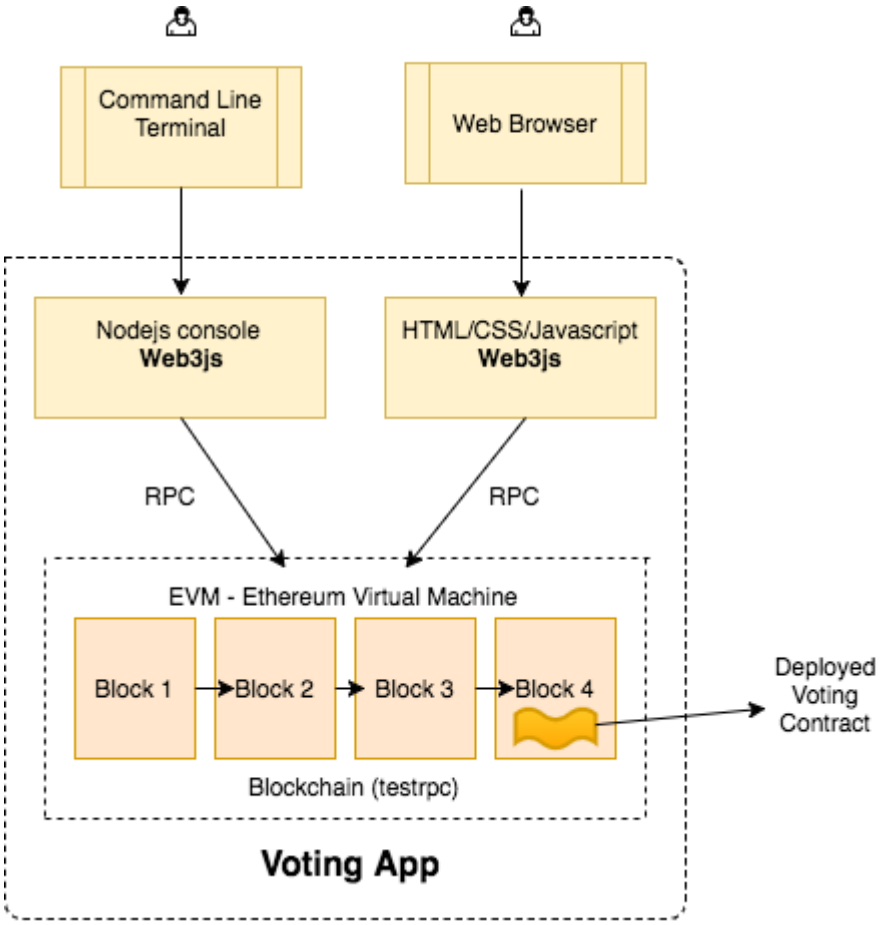
V mém případě je toto zbytečné, pro takto malou aplikaci mi stačí pouze jeden chytrý kontrakt. Zbytečné proto, jelikož za každé další volání jiného kontraktu se zvyšuje cena.

4.5.2 Uživatelský interface

Moje uživatelské interface se skládá z HTML stránky stylované CSS. Jelikož píšu decentralizovanou aplikaci používám také web3js, javascriptovou knihovnu, která mi umožňuje komunikaci s blockchainem.

4.5.3 Transakční interface

Zde jsem se rozhodl využít peněženku Mist a její prohlížeč Mist Navigator. Mist umožnuje mít uloženy všechny osobní informace lokálně na počítači, a komunikaci s blockchainem zařizuje on. Není proto potřeba žádného přihlašování nebo poskytování údajů, vše zůstává plně v rukou návštěvníka. Ten si ve svém Mist prohlížeči zvolí, kterým profilem se chce ke stránce připojit a jaká data jí je ochoten poskytnout.



Obrázek 7 Architektura volební aplikace (Vats, 2018)

Na obrázku 7 je schéma volební aplikace, které je použito i v této aplikaci. V přístupu k aplikaci je možné využít webový prohlížeč nebo shell, ten poskytne propojení s blockchainem, kam ukládáme nové hlasování, nebo volené hlasy, které jsou zapisovány do bloků. Hlavní logika aplikace je uložena v jejím front endu, chytrý kontrakt stejně jako funkce pro zapisování hlasů do logu a jejich počty jsou volány zde.

4.6 Předpoklady pro vývoj

Prostředí

- VS Code
- Solidity
- Truffle
- Mist peněženka a prohlížeč
- Node.js
- Web3js

4.7 Vývoj

V této kapitole se s vámi podělím o zajímavé části kódu z programování. Jsou to především kusy kódu zaměřené na Solidity a web3js, které se starají o komunikaci s blockhainem.

4.7.1 Smart kontrakt

```
pragma solidity ^0.5.0;
contract VoteContract {
    event LogVote(bytes32 indexed proposalHash, address addrs, uint
answerId);
        function vote(bytes32 proposalHash, uint answerId) internal{
        if (msg.value > 0)  {revert();}
        // Log Vote for later calculations
        emit LogVote(proposalHash, msg.sender, answerId);
    }
}
```

Z toho kusu kódu je vidět, jak vypadá jednoduchý kontrakt v solidity. V prvním řádku si deklaruji, jakou verzi solidity používám. Dále si vytvořím event, který má jako

výstup haš hlasovaného kontraktu možnost, odpověď, pro kterou hlasuji a adresu respondenta. Ve funkci „vote“ kontroluji, aby se náhodou neposlal žádný ether, kdyby se tak stalo, všechny hodnoty se vrátí zpět a poplatek je vrácen zpět zasílateli do peněženky. Jinak je proveden event. „LogVote“, ten zapíše hodnoty, které jsou pak později kalkulovány aplikací.

4.7.2 Připrav návrh hlasování

```
// Get the proposal
      propHash = web3.sha3(proposal);

      if (typeof proposal == 'undefined' || proposal == 'null' || proposal
      == '') {
           // No Proposal were found
       } else {
           web3.eth.filter('latest').watch(function(e, res){
               if(!e) {
                   calculateAllVotes();
               }
           });
       }
```

Získání kontraktu probíhá na webu pomocí web3js, jak je vidět v prvním řádku kódu nad textem. propHash je v mém demu hodnota zadaná jako otázka na kterou se ptáme, ta je zahašována pomocí .sha3 funkce. Jakmile získám hodnotu propHash, mohu pomocí metody web3.eth.filter(‘lastest‘) podívat se po změnách v posledním bloku pro dané hlasování. Pokud je sledování blockchainu úspěšné, volám svoji funkci calculateAllVotes(). (ethereum community, 2014)

4.7.3 Načti hlasování

```
// Load the contract, by using his address
    web3.eth.getCode(contractAddress, function(e, r) {
        if (!e) {
            web3.eth.getCode(contractAddress, function(e, r) {
                    if (!e) {
                        // Set up variables from contract
                        ethContract = web3.eth.contract(contractABI);
                        ethVote = ethContract.at(contractAddress);
                    }})
```

```
            // Watch Votes
            if (proposal && proposal.length > 0 && proposal != 'null')
                checkVotes();
        }
    })
```

Načtení kontraktů podle specifikované adresy, ta je obsažena v URL každého kontraktu a je pro něj jedinečná. Po tom, co se úspěšně načte adresa kontraktu, lze její pomocí získat veškeré informace o kontraktu.

4.7.4 Sledování hlasování

```
function checkVotes() {

    // call logVotes event on the contract
    var logAllVotes = ethervote.LogVote({proposalHash: proposalHash},
{fromBlock: startingBlock});

    logAllVotes.watch(function(error, res){
        if (!error) {
            web3.eth.getBalance(res.args.addrs, function(err, balanceInWei){
                // Get the current balance of a voter
                var bal = Number(web3.fromWei(balanceInWei, "finney"));
                voteMap[res.args.addrs] = {balance: bal, support:
res.args.answerId};
                // Check if the current owner has already voted and show it
on their page
                web3.eth.getAccounts(function(e,accounts){
                    if (!e && accounts && accounts[0] == res.args.addrs) {
                        if (res.args.answerId = 1) {
                            // show that this answer was picked
                        } else {
                            // show that different answer was picked
                        }
                    }
                });
               // calculate votes
                calculateAllVotes();
            })
        }
    })
}
```

Funkce pro sledování hlasování, vždy volá event logAllVotes, jelikož je nutné mít zapsané všechny hlasy v uloženy logu, aby je bylo možné spočítat. Zajímavější, ale je zobrazování výsledků na frontend v tomto případě, pokud je vše bez errorů, máme učet a ten má již uloženou jeho adresu v bloku, podle toho, jestli zvolil možnost 1 nebo 2, zobrazí se mu jeho odpověď, a výsledek hlasování např.

4.7.5 Počítání hlasů

```
function calculateAllVotes() {

    for (var a in voteMap) {
        // call the function asynchronously
        web3.eth.getBalance(a, function(e,r) {
            voteMap[a].balance = Number(web3.fromWei(r, 'finney'));
            updateResults()
        });
    };
}
```

Pro zajímavost jsem zde uvedl funkci na počítání hlasů, kde si převádím jednotky ze základních Wei (1 ETH = 1000000000000000000 Wei) do Finney (1 ETH = 1000 Finney).

4.7.6 Nový proposal

```
function createNewProposal() {
    // Create new dynamic URL when creating new proposal
    var createNewProp = document.getElementById('new-proposal');
    var createNewPropLnk = document.getElementById('new-proposal-link');
    createNewPropLnk.href = '?proposal=' + encodeURI(createNewProp.value);
}
```

Tvorba nového hlasování musí vždy vygenerovat jedinečný URL odkaz na toto hlasování. Takový odkaz lze vyrobit velice jednoduše, jakmile uživatel zadá název nového hlasování, zavolá se funkce encodeURI, která enkóduje každou instanci určitých znaků, jedním, nebo více sekvencemi UTF-8 znaky. (MDN Web Docs, 2005) Toto je pak spojeno do promněné createNewProposalLink.href a vloženo zpět na webovou stránku, aby se uživatel mohl prokliknout ke svému hlasování a zároveň měl na něj odkaz.

5 Diskuse

Prototyp aplikace byl úspěšně nasazen na TestNet, testovací část Ethereum sítě, jeho funkcionalita otestována a výsledky zaznamenány. Aplikace nebyla otestována na živém Ethereum blockchainu kvůli výšce nákladů na takovouto operaci. Obě prostředí jsou si nicméně rovna a autor nevidí žádné rozdíly v technickém rozhraní obou blockchainů.

Po dalším rozšíření by bylo možné aplikaci využít celou. Toto rozšíření by si žádalo upravení webové aplikace, resp. designu webové stránky, pro lepší vzhled. Dále rozšířit funkcionalitu chytrého kontraktu. Chytrý kontrakt plnil funkci hlasování a odesílání dat na blockchain, nicméně chybí funkcionalita plného přihlašování uživatelů, a zobrazovaní hlasovacích dat do přehledné podoby, jakožto stylování webových stránek nebylo součástí této bakalářské práce.

V rámci testování byla vytvořena testovací otázka. Pomocí otázky byl vytvořen kontrakt, který byl nasazen na blockchain. Odpovědi byly zaregistrovaný v aplikaci a uloženy. Výsledky hlasování aplikace zobrazila ihned po hlasování. Volební hlas byl správně přiřazen a nedošlo k žádnému pochybení.

Podle autora je mínusem Etherea kurz této kryptoměny, který je v době odevzdání práce zhruba tři tisíce korun, za jednu jednotku[1]. Cena Etheru momentálně stagnuje po nízkém růstu, podle autora se dá v budoucnu předpokládat kontinuální růst, nikoliv pokles cen. Autor zastává tento názor, kvůli rostoucí bázi společností investujících do blockchainu a Etherea. Zejména z finančního sektoru, kde by takovéto transakce mohly způsobit revoluci díky jejich rychlosti a nezpochybnitelnosti.

[1] Kurz vzatý dne 14.3., zdroj:https://www.kurzy.cz/ethereum/

6 Závěr

Práce se zabývala problematikou decentralizované databáze, známé jako blockchain. Popis této technologie byl poskytnut v kapitole 3.1 Blockchain v teoretické části bakalářské práce. Byla popsána historie blockchain technologie, důvod jejího vzniku a její vlastnosti. Další část teoretické bakalářské práce, kapitola 3.2 Ethereum, se věnuje Ethereu, jako platformě pro psaní chytrých kontraktů, je popsáno fungování Etherea, a funkcionalita chytrých kontraktů.

V praktické části, v kapitole 4, je popsán návrh webové aplikace s decentralizovanou databází. V průběhu návrhu aplikace je vytvořeno a otestováno demo, které simuluje zapisování hlasování do blockchain databáze. V Kapitole 4.7 Vývoj je zobrazena a popsána základní funkčnost a stěžejní části aplikace, bez kterých by nebyla funkční. Jsou zde hlavně zobrazeny části kódu důležité pro práci a komunikaci s blockchainem. Cílem této bakalářské práce bylo dokázat, že lze vytvořit hlasovací aplikaci na bázi decentralizované databáze.

Práce by mohla být rozšířena o bližší pohled na praktickou stránku blockchain technologie. Pokud by tato práce sloužila jako předloha pro funkční webovou aplikaci, musela by být rozšířena o další funkcionalitu, která nebyla potřeba v základní funkcionalitě a k zobrazení fungovaní blockchainu ve webové aplikaci.

7 Citovaná literatura

Buterin, V. (2014). ethereum/wiki. [online] GitHub. Available at: https://github.com/ethereum/wiki/wiki/White-Paper#applications [Accessed 16 Feb. 2019].

Zhao, J., Fan, S. and Yan, J. (2016). Overview of business innovations and research opportunities in blockchain and introduction to the special issue. Financial Innovation, 2(1).

WOOD, G. (n.d.). ETHEREUM: A SECURE DECENTRALISED GENERALISED TRANSACTION LEDGER. [online] Gavwood.com. Available at: https://gavwood.com/paper.pdf [Accessed 11 Feb. 2019].

Crosby MA, Pattanayak P, Verma S, Kalyanaraman V (2016) BlockChain Technology: Beyond Bitcoin. Applied Innovation, No. 2, pp. 6–10
Google Scholar

Gates, M. (2017). Blockchain: Ultimate guide to understanding blockchain, bitcoin, cryptocurrencies, smart contracts and the future of money.. 1st ed.

de Kruijff, J. and Weigand, H. (2017). Understanding the Blockchain Using Enterprise Ontology. Advanced Information Systems Engineering, pp.29-43.

En.bitcoin.it. (2019). Bitcoin Wiki. [online] Available at: https://en.bitcoin.it/ [Accessed 13 Jan. 2019].

Bitcoin.org. (2009). Developer Documentation – Bitcoin. [online] Available at: https://bitcoin.org/en/developer-documentation [Accessed 13 Mar. 2019].

Hallion, M. (2018). How to refer to gender-neutral and trans relatives - Quora. [online] Quora.com. Available at: https://www.quora.com/How-do-you-refer-to-gender-neutral-and-trans-relatives-Most-words-like-aunt-uncle-and-niece-nephew-are-binary.

AnySilicon. (2018). The Ridiculously Smart Guide to Developing your own ASIC - AnySilicon. [online] Available at: https://anysilicon.com/ridiculously-smart-guide-developing-asic/ [Accessed 23 Feb. 2019].

Hertig, A. (n.d.). How Do Ethereum Smart Contracts Work? - CoinDesk. [online] CoinDesk. Available at: https://www.coindesk.com/information/ethereum-smart-contracts-work [Accessed 18 Feb. 2019].

Kasireddy, P. (2017). How does Ethereum work, anyway?. [online] Medium. Available at: https://medium.com/@preethikasireddy/how-does-ethereum-work-anyway-22d1df506369 [Accessed 2 Mar. 2019].

Solidity.readthedocs.io. (2016). Solidity — Solidity 0.5.7 documentation. [online] Available at: https://solidity.readthedocs.io/en/latest/ [Accessed 13 Mar. 2019].

Swan, M. (2015). Blockchain. Sebastopol, CA: O'Reilly Media, Inc.

Antonopoulos, A. (2014). Mastering bitcoin. 1st ed. O'Reilly Media.

Kim, K. (2018). Modified Merkle Patricia Trie — How Ethereum saves a state. [online] Medium. Available at: https://medium.com/codechain/modified-merkle-patricia-trie-how-ethereum-saves-a-state-e6d7555078dd [Accessed 7 Mar. 2019].

SEANG, S. and TORRE, D. (2018). Proof of Work and Proof of Stake consensus protocols: a blockchain application for local complementary currencies. [online] Available at: https://gdre-scpo-aix.sciencesconf.org/195470/document [Accessed 7 Mar. 2019].

Buterin, V. (2014). Slasher: A Punitive Proof-of-Stake Algorithm. [online] Blog.ethereum.org. Available at: https://blog.ethereum.org/2014/01/15/slasher-a-punitive-proof-of-stake-algorithm/ [Accessed 10 Mar. 2019].

Truffle Suite. (2019). Truffle Suite | Documentation | Truffle | Overview. [online] Available at: https://truffleframework.com/docs/truffle/overview [Accessed 14 Jan. 2019].

State of the ĐApps Blog. (2018). The ABC of a great DApp Architecture. [online] Available at: https://blog.stateofthedapps.com/the-abc-of-a-great-dapp-architecture-74c70443bc95 [Accessed 23 Jan. 2019].

ethereum community (2014). ethereum/wiki. [online] GitHub. Available at: https://github.com/ethereum/wiki/wiki/JavaScript-API#web3 [Accessed 14 Feb. 2019].

MDN Web Docs. (2005). encodeURI(). [online] Available at: https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/encodeURI [Accessed 26 Jan. 2019].

Blockgeeks.com. (n.d.). [online] Available at: https://blockgeeks.com/guides/what-is-blockchain-technology/ [Accessed 6 Jan. 2019].

Vats, R. (2018). Introduction to D-apps. Part #2. [online] Medium. Available at: https://medium.com/coinmonks/introduction-to-d-apps-part-2-287dc8c3620e [Accessed 3 Feb. 2019].

Ethdocs.org. (2019). Ethereum Homestead Documentation — Ethereum Homestead 0.1 documentation. [online] Available at: http://www.ethdocs.org/en/latest/index.html [Accessed 5 Feb. 2019].

Stephen, F. (2017). Blockchain Technology: Introduction to Blockchain Technology and Its Impact on Business Ecosystem. 1st ed. CreateSpace Independent Publishing Platform.