Primjena okvira pri razvoju softvera temeljenog na lancu blokova

Težak, Dominik

SVEUČILIŠTE U ZAGREBU

FAKULTET ORGANIZACIJE I INFORMATIKE

V A R A Ž D I N

Dominik Težak

Primjena okvira pri razvoju softvera temeljenog na lancu blokova

ZAVRŠNI RAD

Varaždin, 2023.

SVEUČILIŠTE U ZAGREBU

FAKULTET ORGANIZACIJE I INFORMATIKE

V A R A Ž D I N

Dominik Težak

JMBAG: 0016145791

Studij: Informacijski sustavi

Primjena okvira pri razvoju softvera temeljenog na lancu blokova

ZAVRŠNI RAD

Mentor:

Dr. sc. Marko Mijač

Varaždin, studeni 2023.

Dominik Težak

Izjava o izvornosti

Izjavljujem da je moj završni rad izvorni rezultat mojeg rada te da se u izradi istoga nisam koristio drugim izvorima osim onima koji su u njemu navedeni. Za izradu rada su korištene etički prikladne i prihvatljive metode i tehnike rada.

Autor potvrdio prihvaćanjem odredbi u sustavu FOI-radovi

Sažetak

Lanac blokova je decentralizirana, neizbrisiva baza podataka koja olakšava praćenje imovine i zabilježbu transakcija unutar korporativne mreže. Imovina može biti fizička (kuća, automobil, novac ili komad zemlje) ili nematerijalna (patenti, autorska prava, marka proizvoda i intelektualno vlasništvo). Na mreži lanca blokova, praktički sve vrijednosti mogu se zabilježiti i trgovati, smanjujući rizik i povećavajući učinkovitost za sve strane. Informacije su ključne za poslovanje. Najbolje je ako se primaju brzo i točno. Lanac blokova je najbolja tehnologija za dostavu tih informacija jer nudi podatke u stvarnom vremenu, koje je moguće dijeliti i potpuno transparentne, a čuvaju se na neizbrisivoj knjizi promjena i pristup imaju isključivo članovi dozvoljene mreže. Između ostalog, mreža lanca blokova može pratiti narudžbe, plaćanja, račune i proizvodnju. Također, jer svi imaju pristup istoj verziji knjige, možete vidjeti svaki aspekt transakcije od početka do kraja, što povećava vaše povjerenje i otvara nove mogućnosti. Sve navedeno proći će se u ovom završnom radu kako bismo dotaknuli sve značajke aspekte samog lanca blokova. Kako bi se olakšalo korištenje tehnologije lanca blokova i ubrzala implementacija zgodno je koristiti okvire. Okviri pružaju temelj na kojem se može izgraditi softver, obično s određenim programskim jezikom. Najbolje su prilagođeni određenim vrstama zadataka i često su povezani s određenim jezikom. U ovom će radu biti navedeni okviri koji se koriste za sam razvoj tehnologije lanca blokova. Pojedini okviri biti će objašnjeni i istaknut će se razlike između njih. U praktičnom dijelu upotrijebit će se jedan okvir za razvoj lanca blokova i prikazat će se upotreba i svrha okvira koja će dovest do zaključka rada.

Ključne riječi: glavna knjiga; javni/privatni ključ; dokaz o radu; rudarenje; distribuirana struktura podataka; pametni ugovori; decentralizacija; kriptografija; tokeni; lanac blokova

Sadržaj

1. Uvod

Lanac blokova (eng. *blockchain*) budi mnogo pitanja i nejasnosti između ljudi. Kako bi se na pitanja što jednostavnije moglo odgovoriti proći će se svi fundamentalni aspekti koji su nužni kako bi se stvorio lanac blokova i tehnologija iza njega. Jedan od vodećih razloga zašto je tehnologija lanca blokova budućnosti je što omogućuje izbacivanje posrednika(eng. *middleman*) iz elektroničkih transakcija, među ostalim potencijalnim prednostima o kojima će se u nastavku prodiskutirati. Globalni prijenosi plaćanja uključuju puno manje posla i obrade zahvaljujući digitalnoj tehnologiji glavne knjige lanca blokova. Valjanost i sigurnost transakcije se provjeravaju, vrše se konverzije valuta i završavaju druge ključne zadatke koje za naknadu obavljaju treće strane. Naknade za slanje novca često se nazivaju "uslužne naknade" iz razloga što treće strane moraju obaviti određene zadatke kako bi se plaćanja obradila, a plaćaju se prema iznosu plaćanja. U ovom radu će se ponajviše istraživat okviri (eng. *framework*) koji služe za jednostavnije, brže i intuitivnije razvijanje softvera lanca blokova. Da bi se razumjelo kako i zašto koristiti okvire potrebno je objasniti tehnologiju lanca blokova koja je presudna kako bi bilo moguće korištenje okvira.

Rad je podijeljen na dva dijela. U prvom dijelu rada će se teorijski razjasniti kako nastaje lanac blokova i metode koje su potrebne za nastajanje istog. Za jasnije objašnjenje koristit će se primjeri iz stvarnog života. Također, opisat će se važni procesi koje se koriste unutar metoda lanca blokova. Nadalje slijedi teorijsko objašnjenje okvira kao što i svrha korištenja okvira. Predstavit će se razlike od najaktualnijih framework-a koji se koriste u današnje vrijeme prilikom razvijanja softvera za tehnologiju lanca blokova. U praktičnom dijelu rada osmislit će se i izraditi softver temeljen na lancu blokova pomoću framework-a. Cilj praktičnog dijela je istražit primjenu tehnologije lanca blokova i prikazati kako se može u stvarnom životu primijenit na različite načine i za različite situacije kojima se susrećemo u današnje vrijeme.

2. Lanac blokova

Lanac blokova je decentralizirana i distribuirana tehnologija koja služi kao transparentna i sigurna digitalna knjiga za bilježenje transakcija i pohranu informacija. To je revolucionarni koncept koji je privukao značajnu pažnju u raznim industrijama, uključujući financije, opskrbni lanac, zdravstvo itd. U svojoj srži, blockchain je lanac blokova, gdje svaki blok sadrži popis transakcija ili podataka. Ti su blokovi međusobno povezani kronološkim redoslijedom, tvoreći nepromjenjivi zapis svih transakcija koje su se dogodile na mreži i koji je evidentan kao neovlašten. Transakcije se potvrđuju i dodaju u lanac blokova putem mehanizma konsenzusa koji su dogovorili sudionici mreže. Jedna od ključnih značajki lanca blokova je njegova decentralizirana priroda. Umjesto da se oslanja na središnje tijelo ili posrednika za provjeru transakcija, lanac blokova koristi mrežu distribuiranih čvorova, koji se često nazivaju "peer" ili "rudari". Ovi čvorovi rade zajedno kako bi potvrdili legitimnost transakcija, osiguravajući postizanje konsenzusa prije njihovog dodavanja u lanac blokova. Ovaj decentralizirani mehanizam konsenzusa povećava sigurnost, budući da jednom entitetu postaje iznimno teško manipulirati ili mijenjati podatke u lancu blokova. Tehnologija temeljena na lancu blokova nudi nekoliko prednosti. Prvo, pruža transparentnost, budući da je cijela povijest transakcija pohranjena na lancu blokova i mogu joj pristupiti svi sudionici. Ova transparentnost povećava povjerenje i odgovornost unutar mreže. Drugo, lanac blokova nudi nepromjenjivost, što znači da se transakcija jednom kad se zabilježi na lancu blokova ne može promijeniti ili izbrisati, što pruža visoku razinu integriteta i mogućnosti revizije. Drugi značajan aspekt lanca blokova je njegov potencijal za omogućavanje „peer-to-peer“ transakcija bez potrebe za posrednicima. To može pojednostaviti procese, smanjiti troškove i povećati učinkovitost u raznim industrijama. Tehnologija temeljena na lancu blokova nije ograničena na financijske transakcije. Također se može koristiti za pohranu i upravljanje drugim vrstama digitalne imovine, kao što su ugovori, identiteti ili intelektualno vlasništvo. Pametni ugovori, koji su samo izvršivi sporazumi napisani u kodu, mogu se implementirati na lancima blokova, automatizirajući i provodeći ugovorne uvjete.

Međutim, lanac blokova nije bez izazova. Suočava se s problemima skalabilnosti, jer je broj transakcija obrađenih u sekundi relativno ograničen u usporedbi s tradicionalnim sustavima. Osim toga, osiguravanje privatnosti uz održavanje transparentnosti delikatna je ravnoteža koju treba riješiti. Unatoč ovim izazovima, lanac blokova ima golem potencijal da revolucionira brojne industrije pružajući transparentnost, sigurnost i učinkovitost. Kako se tehnologija nastavlja razvijati i sazrijevati, očekuje se da će njezin utjecaj na naš digitalni svijet značajno porasti u nadolazećim godinama. [1]

2.1. Decentralizacija

Moć, kontrola i ovlasti za donošenje odluka raspoređeni su preko mreže ili sustava umjesto da budu koncentrirani u jednom središnjem tijelu. Decentralizacija je temeljno načelo u kontekstu lanca blokova, s ciljem uklanjanja potrebe za posrednicima i osnaživanja pojedinačnih sudionika mreže. Decentralizacija je temeljni princip u tehnologiji lanca blokova. Tradicionalni sustavi oslanjaju se na centralizirane posrednike koji olakšavaju transakcije i potvrđuju informacije, poput banaka ili državnih institucija. Lanac blokova, s druge strane, decentralizira te funkcije korištenjem mreže distribuiranih čvorova, od kojih svaki održava kopiju knjige lanca blokova. [2]

2.1.1. Prednosti decentralizacije

Prednosti decentralizacije lanca blokova uključuju povećanu sigurnost jer ne postoji jedna točka kvara koja je osjetljiva na napade ili manipulaciju. Budući da je cijela povijest transakcija otvorena i sudionici je mogu provjeriti, decentralizacija promiče transparentnost i povjerenje. Omogućuje otpor cenzuri osiguravajući da niti jedno središnje tijelo ne može ograničiti ili kontrolirati pristup mreži. Decentralizacija također promiče inovativnost i jedinstvenost izjednačavanjem uvjeta za sudionike i omogućavanjem raznolikih doprinosa mreži.[3]

2.1.2. Nedostatci decentralizacije

Postoje neke poteškoće i potencijalni nedostaci decentralizacije. Budući da distribuirana mreža sudionika mora postići konsenzus, to može rezultirati sporijim procesima donošenja odluka i poteškoćama u koordinaciji. Skalabilnost može biti problem jer je kapacitet mreže za rukovanje velikom količinom transakcija ograničen. Decentralizirani sustavi također se mogu suočiti s problemima upravljanja jer postizanje konsenzusa o promjenama ili nadogradnjama protokola može biti teško.[3]

2.2. Glavna knjiga

Tehnički okvir i protokoli poznati kao tehnologija distribuirane glavne knjige (eng. Distributed Ledger Tehnology) omogućuje istovremeni pristup, validaciju zapisa i ažuriranje zapisa kroz mrežnu bazu podataka. Lanci blokova se temelje na tehnologiji distribuirane knjige, čija infrastruktura omogućuje korisnicima da vide sve promjene i osobe koje su ih izvršile, smanjuje potrebu za provjerom podataka, osigurava pouzdanost podataka i ograničava pristup samo onima koji ga zaista trebaju. Tehnologija distribuirane glavne knjige omogućuje sigurno i precizno pohranjivanje informacija putem kriptografije. Podacima se može pristupiti putem "ključeva" i kriptografskih potpisa. Nakon što su informacije pohranjene, mogu postati neizbrisiva baza podataka. Distribuirane knjige manje su ranjive na cyber kriminal zbog decentralizacije, privatnosti i enkripcije. Napad mora biti istovremeno pokrenut na svaku kopiju pohranjenu diljem mreže. Dodatno, proces je brži, učinkovitiji i jeftiniji zahvaljujući dijeljenju i ažuriranju zapisa među korisnicima. Distribuirana mreža knjiga pohranjuje kopiju knjige na svakom uređaju povezanom s njom. Čvorovi su ti uređaji, i u mreži može biti proizvoljan broj čvorova. Svi čvorovi istovremeno bilježe sve promjene u knjizi, poput prijenosa podataka između blokova. Svaki čvor objavljuje vlastitu verziju knjige s najnovijim transakcijama jer svaki od njih ima kopiju knjige. Transakcije se zaključuju, kriptiraju i koriste kao temelj za sljedeće transakcije ako mreža utvrdi da je najnovija verzija knjige pouzdana. Na taj način se razvijaju lanci blokova i svaki blok sadrži kriptirane informacije o prethodnom bloku, čime se onemogućuju promjene na njima. [4]

2.3. Kriptografija

Informacije se skrivaju ili kodiraju pomoću kriptografije kako bi ih samo primatelj poruke mogao razumjeti. Još od davnina, poruke se kodiraju pomoću umjetnosti kriptografije, koja se i danas koristi u e-trgovini, bankovnim karticama i računalnim lozinkama. Algoritmi i šifre koje se koriste u suvremenoj kriptografiji, poput šifara sa 128-bitnim i 256-bitnim ključevima za enkripciju, omogućuju enkripciju i dekriptiranje podataka. Moderni kriptografski algoritmi smatraju se gotovo neprobojnima, poput Naprednog standarda enkripcije (eng. Advanced Encryption Standard). Praksa kodiranja podataka kako bi se osiguralo da samo osoba za koju je poruka napisana može pročitati i obraditi informacije je uobičajena definicija kriptografije. Područje kibernetičke sigurnosti poznato kao kriptologija kombinira elemente računalne znanosti, inženjerstva i matematike kako bi proizvelo sofisticirane kodove koji skrivaju pravi značaj poruke.[5]

2.4. Postupak konsenzusa ili sporazuma

U sustavima lanca blokova, mehanizam konsenzusa je program koji olakšava distribuirani konsenzus o trenutnom stanju knjige. Obično se implementira u mreži s velikim brojem korisnika i procesa. Upotreba mehanizama konsenzusa korisna je za distribuirane knjige, lance blokova i kripto valute jer mogu zamijeniti puno sporije ljudsko provjeravanje i verifikaciju.

Postoje različite vrste algoritama mehanizma konsenzusa, od kojih svaki radi prema različitim načelima.

Dokaz rada (eng. Proof of Work) je uobičajeni algoritam konsenzusa koji se koristi u najpopularnijim mrežama kripto valuta poput Bitcoina i Litecoina. Zahtijeva da sudionički čvor dokaže da je obavljeni i predani rad kvalificiran za pravo da dodaju nove transakcije na lancu blokova. Još jedan popularni algoritam konsenzusa je dokaz udjela (eng. Proof of Stake), koji se pojavio kao jeftina zamjena za algoritam za dokaz rada s manjom potrošnjom energije. Uključuje dodjeljivanje sudioničkim čvorovima udjela u održavanju javne knjige temeljem količine virtualnih valutnih tokena koje posjeduju. Nedostatak toga je da potiče zadržavanje umjesto trošenja. Postoje i drugi algoritmi konsenzusa poput dokaza kapaciteta (eng. Proof of Capacity), koji omogućuju dijeljenje memorijskog prostora sudioničkih čvorova u mreži lanca blokova, iako su dokaz o radu i dokaz o udjelu zasigurno najčešći u prostoru lanca blokova. Čvoru se dodjeljuju veća prava za održavanje javne knjige što više memorije ili prostora na tvrdom disku posjeduje. Protokol dokaz o aktivnosti (eng. Proof of Activity) na Decred lancu blokova je hibridni algoritam koji kombinira elemente dokaz o radu i dokaz o udjelu. Dokaz o ispaljenju (eng. Proof of Burn) je protokol za transakcije kripto valuta koji korisnike traži da pošalju male količine kripto valute na nedostupne adrese novčanika, čime se praktički "uništavaju" te adrese.[6]

2.5. Pametni ugovori

Pametni ugovor je samoizvršavajući program koji automatizira radnje potrebne u dogovoru ili ugovoru. Nakon završetka, transakcije su lake za nadziranje i nepovratne. Bez potrebe za centraliziranom vlasti, pravnim sustavom ili vanjskim mehanizmom provedbe, pametni ugovori omogućuju pouzdane transakcije i sporazume između raspršenih, anonimnih strana.

Jednostavne izjave poput "ako/kada...tada" koje se zapisuju u kodu i postavljaju na lancu blokova opisuju kako funkcioniraju pametni ugovori. Kada se utvrdi da su unaprijed određeni uvjeti ispunjeni, mreža računala će provesti akcije. To može uključivati isplatu novca pravim osobama, registraciju vozila, slanje obavijesti ili izdavanje kazni. Kada se transakcija završi,

lancu blokova se ažurira. Kao rezultat toga, transakcija se ne može mijenjati, a samo strane kojima je odobrena dozvola mogu vidjeti rezultat. U pametni ugovor mogu se uključiti sve potrebne uvjete kako bi se sudionicima osiguralo da će zadatak biti uspješno dovršen. Sudionici moraju se složiti oko pravila "ako/kada...tada" koja reguliraju te transakcije, razmotriti sve moguće iznimke i definirati okvir za rješavanje sporova kako bi se uspostavili uvjeti. Sudionici također moraju odlučiti kako će se transakcije i njihovi podaci prikazivati na lancu blokova. Razvojni programer potom može programirati pametni ugovor, iako sve više poslovnih subjekata koji koriste lanac blokova za poslovanje nude predloške, web sučelja i druge online alate kako bi olakšali strukturiranje pametnih ugovora.[7]

2.6. Nepromjenjivost

U kontekstu lanca blokova, neizmjenjivost se odnosi na nemogućnost izmjene ili modificiranja prethodno zabilježenih podataka. Jednom kada se transakcija dodijeli bloku i potvrdi od strane mreže, gotovo je nemoguće promijeniti je.

2.7. Transparentnost

Transparentnost u lancu blokova je otvorenost i vidljivost informacija unutar mreže lanca blokova. Sudionici mogu pristupiti i provjeriti podatke pohranjene na lancu blokova, što je ključna karakteristika tehnologije temeljene na lancu blokova. Svaka osoba s dozvoljenim pristupom lancu blokova može provjeriti podatke i detaljno ih analizirati zahvaljujući transparentnosti.

Primjer transparentnosti u tehnologiji temeljenoj na lancu blokova je otvorenost određenih mreža lanca blokova poput Bitcoina i Ethereuma. U tim mrežama, svatko može vidjeti i provjeriti povijest transakcija i povezane informacije, uključujući adrese novčanika i iznose transakcija. Postoji visok stupanj transparentnosti i odgovornosti jer svatko može vidjeti transakcije i njihove pojedinosti. Zahvaljujući mogućnosti neovisne provjere točnosti i integriteta podataka na lancu blokova, sudionici mreže imaju koristi od ove otvorenosti u izgradnji povjerenja.

2.8. Sa dozvolom naspram bez dozvole za lanac blokova

Najznačajnije odstupanje od izvornog koncepta lanca blokova bilo je pokušaj prilagodbe tehnologije suvremenim kontekstima usmjerenim na sigurnost.

Jer, iako je javni oblik lanca blokova bio tehnički siguran protiv napada bruteforce i dvostrukog trošenja, nije bila tehnologija na kojoj se mogu trgovati osjetljive informacije jer bi te informacije bile vidljive svima u mreži. Kako bi se riješili problemi nastale su dvije vrste organizacija. Jedna u kojoj sustav zahtjeva dozvolu, koristi se za pseudo-decentralizirane lance blokova koji se koriste za upravljanje podacima u privatnim poslovnim kontekstima. A druga ona koja ne zahtijeva dozvolu i koja se koristi za javno radikalno decentralizirane. U nastavku će biti objašnjene neke prednosti i nedostaci istih.[8]

2.8.1. Potreba dozvole

Potreba dopuštenja, kojima obično upravlja kontrolni subjekt, omogućuje relativno jednostavnu implementaciju nadogradnje. Takvi lanci blokova mogu se prilagoditi određenim funkcijama, optimizirajući njihovu učinkovitost. Prilagodba je jednostavna ako se zahtjevi promijene. Ovlašteni operateri lanca blokova mogu definirati odgovarajuću razinu transparentnosti na temelju slučaja korištenja mreže. Selektivno sudjelovanje, pristup lancu blokova ograničen je na pozvane sudionike, što omogućuje kontrolu nad tim tko se može, a tko ne može pridružiti.

2.8.2. Bez potrebe dozvole

Jedna od prednosti organizacije bez potrebe dozvole jest da ima potencijal za decentralizaciju. Važno je napomenuti kako nije slučaj da je svaki dozvoljeni lanac blokova decentraliziran, ali obično sadrže potencijal da budu. Uz dovoljnu količinu resursa svatko može sudjelovati u mehanizmu konsenzusa ili koristiti dozvoljenu mrežu. Nadalje korisnici sadrže pravo za sudjelovanju i donošenje odluka u mreži, što dovodi do glasovanja koje naposljetku prouzrokuje proširenje mreže i dodavanje vilica. Jednostavan pristup je također prednost jer kako se radi o mreži koja je dozvoljena svima, svatko može stvoriti novčanik/račun i pridružiti se mreži.[9]

2.9. Tokenizacija

Proces stvaranja i predstavljanja digitalne imovine ili tokena na mreži lanca blokova naziva se tokenizacija. Ovi tokeni mogu predstavljati različitu imovinu, uključujući valute, vrijednosne papire, nekretnine, intelektualno vlasništvo, pa čak i fizičku imovinu kao što su umjetnička djela ili roba. Tokenizacija omogućuje da ta imovina bude digitalno predstavljena, praćena i sigurno prenijeta na lanac blokova. Tokenizacijom pretvaramo imovinu stvarnog svijeta u digitalne tokene koji su jednostavni za identificiranje i za zapisivanje u lanac blokova. Svaki od tih tokena predstavlja određenu imovinu ili dio imovine. Prethodno nelikvidna imovina, kao što su umjetnička djela ili nekretnine, mogu se podijeliti u manje jedinice koje omogućuju lakšu kupnju, prodaju i djelomično vlasništvo. Ovim postupkom povećava se likvidnost tržišta kao što se i povećava krug ulagača kako je moguće ulaganje u manjim svotama. Tokeni na lancu blokova mogu imati programiranu funkcionalnost putem pametnih ugovora. To znači da tokeni mogu imati unaprijed definirana pravila i uvjete pridružene sebi, omogućujući automatizaciju određenih procesa, kao što su raspodjela dividendi, glasačka prava ili usklađenost s regulatornim zahtjevima. Tokenizacija na mreži lanca blokova može olakšati interoperabilnost, dopuštajući jednostavan prijenos ili korištenje tokena na različitim platformama i aplikacijama unutar ekosustava lanca blokova. Popularna metoda korištenja tokena jest upravo za provođenje kampanja skupnog financiranja ili početnih ponuda novčića (ICO). Startupi ili projekti mogu izdavati tokene kao način prikupljanja sredstava, a investitori mogu sudjelovati u tim ponudama stjecanjem tih tokena. Sve u svemu, tokenizacija u lancu blokova nudi način za digitalizaciju i predstavljanje imovine stvarnog svijeta na sigurnoj i transparentnoj decentraliziranoj mreži, otključavajući nove mogućnosti za vlasništvo imovine, trgovanje i financijske inovacije.[10]

2.10. Skalabilnost

Skalabilnost u lancu blokova odnosi se na sposobnost mreže lanca blokova da obrađuje sve veći broj transakcija ili korisnika bez žrtvovanja performansi. Kritični je aspekt tehnologije lančanih blokova da se usvoji u velikoj mjeri i da se široko koristi.

Tradicionalni lanci blokova, kao što su Bitcoin i Ethereum, imaju inherentna ograničenja u pogledu brzine obrade transakcija i kapaciteta mreže, zbog čega je skalabilnost lanca blokova složen izazov. Ta su ograničenja uzrokovana čimbenicima kao što su veličina bloka, vrijeme potvrde i algoritmi konsenzusa. Jedan od načina poboljšanja skalabilnosti je povećanje veličine bloka, što omogućuje uključivanje više transakcija u svaki blok. Međutim, veći blokovi zahtijevaju više računalnih resursa i kapaciteta za pohranu, što otežava pojedinačnim

čvorovima sudjelovanje u mreži. To također može dovesti do centralizacije, jer samo čvorovi sa značajnim resursima mogu podnijeti povećane zahtjeve. Algoritmi konsenzusa, kao što su Proof of Work (PoW) i Proof of Stake (PoS), važni su za sigurnost lanca blokova, ali mogu ograničiti skalabilnost. Skalabilnost se može poboljšati optimizacijom ovih algoritama ili razvojem novih mehanizama konsenzusa. Na primjer, Ethereum 2.0 prijeći će s PoW na PoS, što će povećati propusnost mrežnih transakcija. Protokoli sloja 2 izgrađeni su na temelju postojećih lanaca blokova kako bi se rasteretila neka obrada transakcija s glavnog lanca blokova. Državni kanali, bočni lanci i kanali plaćanja, primjerice, omogućuju brže i jeftinije transakcije smanjenjem opterećenja glavnog lanca. Rješenja sloja 2 mogu značajno poboljšati skalabilnost obradom velike količine transakcija izvan lanca dok istovremeno iskorištavaju sigurnost temeljnog lanca blokova. Nadalje usitnjavanje je tehnika koja uključuje dijeljenje mreže lanca blokova na manje particije koje se nazivaju krhotine. Svaka krhotina može samostalno obrađivati svoje transakcije i pametne ugovore, učinkovito povećavajući kapacitet mreže za paralelno rukovanje s više transakcija. Usitnjavanje zahtijeva pažljiv dizajn i koordinaciju kako bi se održala sigurnost i osigurala pravilna provjera transakcija kroz krhotine. Neke se transakcije mogu izvršiti izvan glavnog lanca blokova i kasnije namiriti u lancu. Transakcije izvan lanca smanjuju opterećenje transakcijama u lancu, poboljšavajući skalabilnost. Kanali plaćanja, kao što je Lightning Network za Bitcoin, omogućuju brze i jeftine transakcije izvan lanca, pogodne za mikro transakcije ili česte prijenose malih vrijednosti. Rješenja interoperabilnosti imaju za cilj povezati različite mreže lanca blokova, omogućujući im komunikaciju i dijeljenje podataka. Omogućavanjem besprijekornog prijenosa sredstava ili informacija između lanaca blokova, interoperabilnost može ublažiti ograničenja skalabilnosti raspodjelom transakcijskog opterećenja na više mreža.

Važno je napomenuti da različite platforme temeljene na lancu blokova i projekti mogu usvojiti različita rješenja skalabilnosti na temelju svojih specifičnih zahtjeva i prioriteta. Postizanje skalabilnosti u lancu blokova često uključuje kompromis između propusnosti, sigurnosti, decentralizacije i upotrebljivosti, a postizanje prave ravnoteže ključno je za održive i učinkovite mreže lanca blokova.[11]

2.11. Komunikacija između mreža lanca blokova

Lanci blokova mogu međusobno komunicirati putem različitih metoda i protokola, omogućujući interoperabilnost i razmjenu podataka ili imovine između različitih mreža lanca blokova.

Jedan od uobičajenih pristupa su Cross-chain mostovi koji su specijalizirani protokoli ili platforme koje olakšavaju komunikaciju između različitih lanca blokova. Oni djeluju kao posrednici, omogućujući prijenos tokena ili imovine između zasebnih lanaca. Ovi mostovi obično zahtijevaju konsenzus sudjelujućih čvorova ili validatora iz oba lanca blokova za potvrdu i provjeru valjanosti transakcija. [12]

Atomic swapovi omogućuju peer-to-peer transakcije između različitih lanca blokova bez potrebe za posrednicima. Ova metoda koristi pametne ugovore kako bi se osigurala istovremena i nepovratna razmjena imovine. Atomske zamjene oslanjaju se na kriptografske tehnike za jačanje povjerenja i sprječavanje prijevara tijekom procesa među lančane transakcije.[13]

Bočni lanci su zasebni lanci blokova koji su povezani ili vezani za glavni lanac blokova. Oni rade paralelno i omogućuju prijenos imovine ili podataka između glavnog lanca i bočnog lanca. Bočni lanci mogu ponuditi veće brzine transakcija ili uvesti specifične funkcionalnosti uz zadržavanje kompatibilnosti i interoperabilnosti s glavnim lancem blokova.[14]

Zatim postoje protokoli interoperabilnosti. Nekoliko protokola i projekata usredotočeno je na razvoj standarda i okvira za interoperabilnost lanca blokova. Ovi protokoli uspostavljaju zajedničke komunikacijske protokole, formate podataka i mehanizme konsenzusa kako bi se omogućila besprijekorna interakcija između različitih mreža lanca blokova. Primjeri uključuju Polkadot, Cosmos i Interledger Protocol (ILP).[15]

Oracles su usluge ili međuprogrami koji pružaju vanjske podatke pametnim ugovorima na lancu blokova. Djeluju kao mostovi između lanca blokova i informacija ili događaja iz stvarnog svijeta. Oracles može omogućiti komunikaciju između lanaca blokova prenošenjem podataka iz jednog lanca blokova u drugi, omogućujući im razmjenu informacija i pokretanje radnji na temelju vanjskih uvjeta.[16]

HTLC (Hashed Timelock Contact) je mehanizam koji omogućuje sigurne transakcije između lanca blokova korištenjem vremenskih zaključavanja i hash funkcija. Omogućuje sudionicima da predlože i izvrše među lančane transakcije unutar određenog vremenskog okvira, osiguravajući atomarnost i sigurnost procesa.[17]

Interoperabilnost između lanca blokova aktivno područje istraživanja i razvoja. Različiti projekti i inicijative istražuju inovativna rješenja za poboljšanje komunikacije i interoperabilnosti

između lanca blokova, s ciljem stvaranja povezanijeg i besprijekornog decentraliziranog ekosustava.

2.12. Privatni i javni ključ

Privatni ključ i javni ključ kriptografski su elementi u tehnologiji lanca blokova koji su ključni za osiguravanje sigurnosti, autentifikacije i povjerljivosti. Privatni ključ je povjerljivi podatak koji se generira nasumično. U kontekstu lanca blokova, privatni ključ povezan je s određenim korisnikom ili entitetom. Služi kao digitalni potpis u slučajevima kada korisnik želi potpisati transakciju ili poruku na lancu blokova. U tom trenutku generira se privatni ključ koji nam predstavlja jedinstveni digitalni potpis. Taj potpis je dokaz da transakcija ili poruka potječe od korisnika koji posjeduje taj odgovarajući privatni ključ. Još jedna svrha privatnog ključa upravo je dešifriranje. Ako lanac blokova koristi tehnike šifriranja, privatni ključ se koristi za dekriptiranje šifriranih podataka namijenjenih korisniku. Ovo je posebno važno u aplikacijama temeljenih na lancu blokova koje imaju usmjerenje na privatnost. Ključno je čuvati privatni ključ povjerljivim i sigurnim jer svatko tko mu ima pristup može lažno predstavljati vlasnika i steći kontrolu nad njegovom digitalnom imovinom ili obavljati radnje u njegovo ime.

U suprotnom javni ključ se matematički izvodi iz odgovarajućeg privatnog ključa pomoću kriptografskih algoritama. Izvodi se na takav način da je neizvedivo odrediti privatni ključ samo iz javnog ključa. Javni ključ, kao što ime sugerira, javno se dijeli i povezan je s određenim korisnikom ili entitetom. Osnovne svrhe javnog ključa su provjera i enkripcija. Kada korisnik primi digitalno potpisanu poruku ili transakciju, može koristiti javni ključ povezan s pošiljateljem za provjeru autentičnosti i integriteta potpisa. Time se osigurava da je poruka ili transakcija doista potpisana privatnim ključem koji odgovara javnom ključu. U slučaju da korisnik želi poslati šifriranu poruku ili podatke drugom korisniku, može koristiti javni ključ primatelja za šifriranje podataka. Nakon šifriranja, samo primatelj s odgovarajućim privatnim ključem može dekriptirati i pristupiti izvornom sadržaju. Ukratko, javni ključevi otvoreno se dijele i mogu se objaviti na lancu blokova ili učiniti dostupnima putem različitih kanala bez ugrožavanja sigurnosti privatnog ključa.[18]

2.13. Vilica

Softver vilica (eng. fork) događa se kada se softver kopira i mijenja. Izvorni projekt nastavlja, ali sada je jasno odvojen od novog, koji ide drugačijim putem. Pretpostavimo da se tim vaše omiljene web stranice o kripto valutama nije složio o daljnjem postupanju. Jedan član tima može odlučiti replicirati stranicu na drugoj domeni. Međutim, u budućnosti bi objavljivali različite vrste sadržaja od originalne verzije.

Postoje dvije vrste vilica to su meka vilica (eng. Soft fork) i tvrda vilica (eng. Hard fork). Razlike su i tome što je meka vilica unazad kompatibilna nadogradnja na protokol lanca blokova. Implementiran je na način da čvorovi koji pokreću novi softver i dalje mogu komunicirati s čvorovima koji pokreću stariju verziju. U mekoj vilici ova pravila ili promjene osmišljene su tako da budu restriktivnije od prethodnih pravila. Blokovi ili transakcije koje se pridržavaju novih pravila i dalje vrijede prema starim pravilima. Stariji čvorovi mogu nastaviti raditi na mreži bez ikakvih problema, sve dok se pridržavaju starih pravila konsenzusa. U scenariju meke vilice, većina hash snage mreže i sudionika dobrovoljno prihvaćaju novi softver, što rezultira glatkim prijelazom. Međutim, čvorovi koji pokreću stariju verziju možda neće moći u potpunosti iskoristiti nove značajke ili poboljšanja uvedena nadogradnjom.

Međutim, tvrda vilica je nekompatibilna nadogradnja protokola lanca blokova koja nije kompatibilna s prethodnim verzijama. Uvodi značajne promjene koje nisu kompatibilne s prethodnom verzijom. U tvrdoj vilici nova pravila ili promjene osmišljene su tako da budu popustljivije ili opširnije od prethodnih pravila. Blokovi ili transakcije koje se pridržavaju novih pravila smatraju se nevažećima prema starim pravilima. Čvorovi koji pokreću stariju verziju ne mogu prepoznati ili potvrditi nove blokove ili transakcije, što dovodi do odstupanja u povijesti lanca blokova. Tvrda vilica stvara dva odvojena lanca: jedan prema starim pravilima, a drugi prema novim pravilima. To često dovodi do stvaranja nove kripto valute ili tokena povezanog s novim lancem. Sudionici moraju odabrati koju verziju lanca blokova koju žele podržati nadogradnjom svog softvera u skladu s tim. Primjeri tvrdih vilica uključuju stvaranje Bitcoin Casha (BCH) iz Bitcoina (BTC) ili Ethereum Classic (ETC) iz Ethereuma (ETH). [19]

3. Okviri za razvoj lanca blokova

Okvir u razvoju softvera je poput gotove strukture na kojoj možete izgraditi svoj softver. Djeluje kao temelj i pošteđuje od potrebe da počinjemo projekte ispočetka. Okviri su obično dizajnirani za određene programske jezike i služe različitim zadacima. Da povučemo analogiju, zamislite da gradite kuću. Mogli biste sami mukotrpno postaviti temelje i uokviriti kuću, ali to bi oduzelo mnogo vremena i truda. Međutim, ako su te temeljne zadatke već dovršili vješti kućni majstori, to bi vam uštedjelo znatne resurse. Isto tako, u razvoju softvera okvir ispunjava sličnu svrhu. Kreirali su ga i testirali iskusni programeri i inženjeri softvera, pružajući pouzdanu i čvrstu početnu točku. Međutim, baš kao što kuća nije potpuna samo svojim okvirom, softverski okvir je samo početna točka. Da bi bio funkcionalan, potrebno je dodavanje funkcija i značajka više razine.[20]

Specifično okviri za razvoj lanca blokova pružaju unaprijed izgrađene predloške, biblioteke i alate koji pojednostavljuju proces razvoja aplikacija lanca blokova. Programeri mogu iskoristiti ove resurse za bržu i učinkovitiju izradu aplikacija. Najbolje prakse i obrasci dizajna za razvoj lanca blokova često su uključeni u okvire za razvoj lanca blokova. To pomaže u osiguravanju pridržavanja aplikacija izrađenih pomoću ovih okvira industrijskih standarda i sigurnosnih smjernica. Neki okviri nude alate za testiranje i simulaciju mreže lanca blokova, omogućujući programerima da procijene izvedbu i ponašanje svoje aplikacije u različitim scenarijima.

Neki od razloga korištenja okvira su sljedeći:

- Brzi razvoj: Okviri pružaju unaprijed izgrađene komponente i isječke koda, ubrzavajući vrijeme razvoja.
- Dosljednost: Okviri često provode standarde i obrasce kodiranja, što rezultira dosljednijim kodom u cijelom projektu.
- Zajednica i podrška: Popularni okviri imaju velike zajednice, što znači više resursa, dokumentacije i pomoći dostupnih na mreži.
- Sigurnost: mnogi okviri uključuju sigurnosne značajke i najbolju praksu, što pomaže u smanjenju ranjivosti u vašoj aplikaciji.
- Skalabilnost: Okviri su dizajnirani za rješavanje problema skalabilnosti, omogućujući aplikacijama da rastu bez velikih promjena arhitekture.

3.1. Okviri za Ethereum

3.1.1.Hardhat

Hardhat je fleksibilno okruženje za razvoj i alat za izvođenje zadataka namijenjen Ethereum projektima. Pojednostavljuje razvoj Ethereum pametnih ugovora i decentraliziranih aplikacija pružajući alate za kompilaciju pametnih ugovora, automatsko testiranje i implementaciju projekata. Hardhat povećava produktivnost razvojnih programera podržavajući prilagodljive zadatke i rješavanje problema. Upravlja konfiguracijama Ethereum mreža, omogućujući testiranje na različitim mrežama. Budući da je Hardhat proširiv, programeri mogu integrirati dodatke i prilagoditi njegove mogućnosti. Popularan je za stvaranje protokola decentraliziranog financiranja (DeFi), Web3 aplikacija i drugih stvari. Hardhat pojednostavljuje proces razvoja pružajući robusni skup značajki i integrirajući se s Ethereum alatima, što ga čini popularnim izborom među razvojnim programerima za razvoj lanca blokova. [21]

3.1.2. Foundry

Napisan u programskom jeziku Rust, Foundry je osmišljen da bude pristupačan, jednostavan za instalaciju i ne zahtijeva kompliciranu konfiguraciju ili biblioteke trećih strana. Sveobuhvatan je skup alata za razvoj i implementaciju decentraliziranih aplikacija temeljenih na Ethereumu. Namijenjen je olakšavanju stvaranja i implementacije sigurnih i učinkovitih pametnih ugovora za razvijatelje svih razina. Mogućnost korištenja Foundry-a za pisanje pametnog ugovora u Solidityju, njegovu kompilaciju, implementaciju na Ethereum lancu blokova te interakciju s njim.[22]

3.2. Okviri za Solanu

3.2.1.Tatum

Tatum je sveobuhvatna razvojna platforma za razvoj lanca blokova koja olakšava integraciju tehnologije lanca blokova u aplikacije. Tatum pojednostavljuje kompleksne procese pružajući skupove API-ja i alata koji omogućuju programerima jednostavno interagiranje s različitim mrežama lanaca blokova. Tatum pruža upravljanje novčanicima, omogućavajući sigurno pohranjivanje kriptovaluta, kao i tokenizaciju, koja omogućava stvaranje i upravljanje prilagođenim tokenima za različite svrhe. Platforma također podržava implementaciju i

interakciju pametnih ugovora, omogućavajući programerima korištenje samopokretnih ugovora za automatizaciju sporazuma. Tatum dodatno olakšava plaćanja kriptovalutama, čineći jednostavnim za tvrtke da prihvate digitalne aktive. Značajke upravljanja identitetom daju prioritet sigurnosti, osiguravajući zaštitu transakcija i korisničkih podataka na lancu blokova. Tatum, sa svojim korisnički prijateljskim sučeljem i snažnim značajkama, omogućava programerima da iskoriste potencijal lanca blokova bez složenosti koja često prati njegovu implementaciju.[23]

3.3. Okviri za Polygon

3.3.1. Curvegrid

Curvegrid nudi MultiBaas, rješenje za srednji sloj lanca blokova, Web3 startupima. Ovaj alat za razvoj softvera omogućuje programerima korištenje poboljšanog ABI-ja (sučelja binarnog programa) platforme, olakšavajući stvaranje visokoučinkovitih decentraliziranih aplikacija. Programeri mogu koristiti MultiBaas API kako bi dodali nove značajke pametnim ugovorima koji već postoje te stvorili nove pametne ugovore prema potrebi. MultiBaas je sveobuhvatni skup alata za API i web-sučelje na temelju kojeg imate potpunu kontrolu nad svojom web3 aplikacijom. Sadrži i opciju pojednostavljenja svojeg radnog procesa i prebacivanja što god želite od kompleksnosti ili koliko malo želite. Pruža učinkovito pristupanje podacima iz novih ili već implementiranih pametnih ugovora u svojim aplikacijama na prednjoj strani ili na strani poslužitelja.[24]

3.3.2. Ape

Ape je okvir za razvoj pametnih ugovora koji omogućuje kompilaciju, implementaciju, testiranje i uklanjanje pogrešaka u vašim projektima. Koristi modularni pristup za pomoć programerima u upravljanju i automatizaciji ponavljajućih zadataka svojstvenih izradi pametnih ugovora, omogućujući korisnicima da izgrade i integriraju vanjske dodatke za dodavanje funkcionalnosti. Kao razvojni programer, možete odabrati koje ćete dodatke koristiti. Ape dolazi unaprijed instaliran s dodacima koji prema zadanim postavkama podržavaju razvoj Ethereuma. Ape-ov ekosustav dodataka kojima upravlja ApeWorX i dodataka zajednice koji se stalno širi omogućuje vam stvaranje prilagođenog razvojnog iskustva. Podržava jezike

Solidity, Vyper i Cairo, ekosustave temeljene na EVM-u poput Polygona, Fantoma i Avalanchea, lance koji nisu EVM (upravljeni zarađenom vrijednošću) poput StarkNeta, pružatelje usluga poput Chainstacka i brojne druge alate ekosustava poput Etherscan Tokenlists.[25]

3.4. Okviri za Avalanche

3.4.1. Robot Framework Solidity Testing Toolkit

Robot Framework Solidity Testing Toolkit je set alata za višelančano testiranje koji uključuje automatizaciju vođenu ključnim riječima za lokalno testiranje pametnih ugovora kompatibilnih s EVM-om (upravljanje zarađenom vrijednošću), njihovu implementaciju korištenjem više lančanog pristupa i pružanje osnovnih sastavnih dijelova za stvaranje botova za praćenje događaja u lancu blokova s jednostavnim ključnim riječima za automatizaciju. Ovaj projekt kombinira nekoliko popularnih Web3 razvojnih alata (Hardhat, Ethers.js, Truffle, Web3.js i tako dalje) i integrira ih s Robot Framework RPA alatom. Ovo je stvoreno kako bi se pokazalo da postoji više pristupa testiranju vezanom uz Web3, decentraliziranih ili aplikacija temeljeni na lancu blokova, te kako bi se potaknulo QA stručnjake da se ne plaše ovih uzbudljivih tehnologija.[26]

4. Korištenje okvira Truffle

U proteklom poglavlju prikazali smo nekoliko okvira koji nam služe za pomoć kod razvijanja tehnologije lanca blokova. Za bolje razumijevanje samih okvira za razvoj tehnologije lanca blokova u ovome dijelu uz pomoć okvira Truffle, editora koda VsCode-a i aplikacije Ganache, korak po korak prikazani će biti postupci kako kreirati, sastaviti i postaviti pametni ugovor na lokalni lanac blokova uz pomoć Truffle-a i Ganache-a.

4.1. Što je Truffle?

Truffle je popularni razvojni okvir i skup alata koji se široko koriste u ekosustavu lanca blokova, posebno na Ethereum platformi. Glavna svrha Truffle-a je pojednostaviti i ubrzati proces razvoja decentraliziranih aplikacija i pametnih ugovora. Pruža programerima dobro

definiranu strukturu projekta koja olakšava organizaciju koda i upravljanje ovisnostima. Osim toga, nudi integrirani kompilator Solidity, omogućavajući programerima pisanje pametnih ugovora na Solidity programskom jeziku te njihovu pretvorbu u bajtkod koji može izvoditi Ethereum virtualna mašina. Jedna od ključnih komponenti Truffle-a je njegov ugrađeni okvir za testiranje. Ovaj okvir omogućava programerima pisanje automatiziranih testova za svoje pametne ugovore, čime se osigurava njihova ispravna funkcionalnost i sigurnost. Implementacija pametnih ugovora može biti izazovna, ali Truffle olakšava ovaj proces automatskim zadacima za implementaciju. Programeri mogu lako migrirati svoje ugovore na različite Ethereum mreže, uključujući glavnu mrežu, testne mreže i privatne lance. Interaktivna konzola Truffle-a pruža programerima razvojno okruženje u kojem mogu integrirati sa svojim pametnim ugovorima u stvarnom vremenu. Ovo olakšava proces ispravljanja grešaka i testiranja funkcionalnosti ugovora. Truffle također podržava upravljanje različitim mrežama lanca blokova i omogućava programerima konfiguraciju tih mreža za implementaciju svojih ugovora. Osim toga, Truffle se integrira s alatima za upravljanje paketima, kao što je npm, čime olakšava upravljanje ovisnostima projekta i dijeljenje decentraliziranih aplikacija s zajednicom. Također, Truffle nudi proširivi sustav dodataka koji programerima omogućava prilagodbu okvira prema specifičnim zahtjevima projekta. Opsežna dokumentacija i aktivna zajednica pružaju dodatnu podršku programerima koji koriste Truffle, čineći ga privlačnom opcijom za razvoj decentraliziranih aplikacija i pametnih ugovora na Ethereum platformi. Uz svoje moćne alate i intuitivno sučelje, Truffle je neprocjenjiv resurs za razvojnu zajednicu lanca blokova.[27]

4.2. Što je Ganache?

Ganache je popularan alat za razvoj lanca blokova, posebno među Ethereum developerima. On djeluje kao emulator lokalnog lanca blokova, pojednostavljujući i ubrzavajući razvoj i testiranje decentraliziranih aplikacija i pametnih ugovora. Ganache pruža korisnički prijateljsko grafičko sučelje (GUI) koje omogućava developerima stvaranje i upravljanje privatnim Ethereum testnim mrežama, s mogućnošću prilagodbe parametara kao što su limiti za gas i vremena između blokova. Ovo lokalno okruženje ključno je za testiranje i otklanjanje grešaka u pametnim ugovorima bez troškova i vremena povezanih s njihovim implementiranjem na Ethereum mainnetu ili testnim mrežama. Ukratko, ovaj ugovor pruža jednostavan način za pohranjivanje i dohvaćanje nepotpisane (eng. unsigned) vrijednosti cijelog broja. Vrijednost možete postaviti pomoću funkcije setter i dohvatiti je pomoću funkcije getter. Ganache podržava Ethereumovu mrežu usmjerenu na razvoj, što znači da programeri

mogu testirati Ether, pregledavati zapisnike transakcija i pregledavati promjene stanja lanca blokova u stvarnom vremenu. To pomaže u otkrivanju i rješavanju pogrešaka, kao i funkcionalnosti i sigurnosti pametnih ugovora. Ganache se također integrira s popularnim razvojnim alatima kao što su Truffle i Remix, što ga čini bitnom komponentom Ethereum razvojnog skupa. To je koristan alat za programere koji žele učinkovito stvarati, testirati i implementirati pametne ugovore i decentralizirane aplikacije, smanjujući vrijeme i troškove razvoja dok poboljšavaju kvalitetu i pouzdanost koda. Zaključno, Ganache je kritičan alat koji Ethereum programerima omogućuje stvaranje robusnih i sigurnih aplikacija lanca blokova.

4.3. Što je VSCode?

Microsoftov Visual Studio Code (VSCode) popularan je, besplatan uređivač koda otvorenog koda. Njegova svestranost, proširivost i robusne značajke doveli su do širokog prihvaćanja među programerima, što ga čini idealnim izborom za razne programske jezike i razvojne zadatke. Korisničko sučelje VSCode-a je pojednostavljeno i vrlo prilagodljivo, što omogućuje programerima da prilagode uređivač svojim specifičnim preferencijama tijeka rada. Podržava širok raspon programskih jezika putem proširenja dostupnih na VSCode tržištu. Ova proširenja dodaju isticanje sintakse, dovršavanje koda, otklanjanje pogrešaka i druge značajke specifične za jezik, što ga čini svestranim alatom za širok raspon razvojnih zahtjeva. Jedna od najznačajnijih značajki VSCodea je njegova integrirana podrška za kontrolu verzije Git, koja pojednostavljuje suradnju i upravljanje kodom. Također pruža robusno iskustvo otklanjanja pogrešaka, s podrškom za različita vremena izvođenja i jezike. Ekosustav proširenja u VSCodeu značajna je snaga, omogućujući programerima da prilagode svoje razvojno okruženje pomoću alata, tema i integracija koje zadovoljavaju njihove potrebe. Lagan je, responzivan i višeplatformski, pa radi na Windowsima, macOS-u i Linuxu.

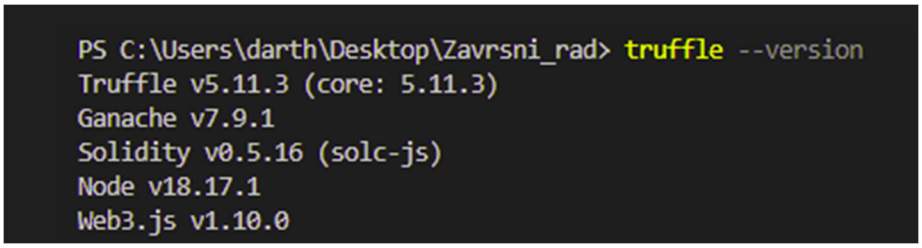
4.4. Implementacija

Za upotrebe prikazivanja mogućnosti okvira Truffle potreban je editor koda VSCode. Pri pokretanju instaliranog editora potrebno je kreirati i otvoriti novu mapu koja je predviđena za inicijalizaciju Truffle-a. Na temelju ispod priloženih slika bit će prikazan postupak nužan za uspješno kreiranje, kompilaciju i postavljanje pametnog ugovora na lokalni lanac blokova.

Za instalaciju Truffle-a potrebno je otvaranje terminala u koji se upisuje komanda: npm install -g truffle.

Slika 1. Instalacija Truffle-a

Ukoliko je smo nesigurni je li instalacija uspješno prošla u terminal upisujemo komandu: truffle - -version.

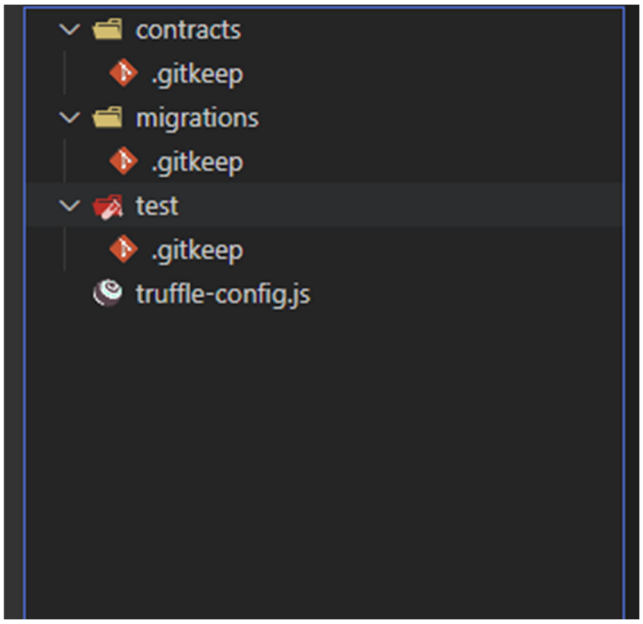


Slika 2. Provjera instalacije

Ako nakon upisane komande truffle - -version dobivamo prikaz kao na slici 2. uspješno je instaliran Truffle. Nadalje su prikazane verzije za ostale alate ili programske jezike koju su na raspolaganju. Zatim upisom komande truffle init kreirat će se sve potrebne mape koje služe za kreiranje, kompilaciju i testiranje pametnog ugovora. Na slici 3. prikazan je pogled iz terminala, a na slici 4. pogled u datoteku koja je prethodno bila kreirana.

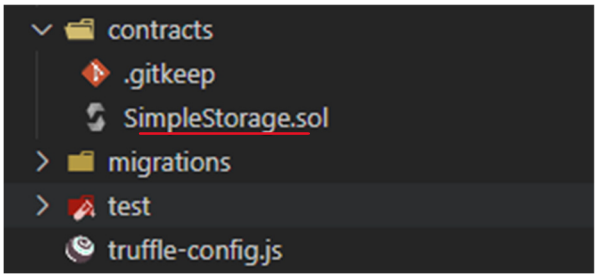
```
PS C:\Users\darth\Desktop\Zavrsni_rad> truffle init

Starting init...
================

> Copying project files to C:\Users\darth\Desktop\Zavrsni_rad

Init successful, sweet!

Try our scaffold commands to get started:
  $ truffle create contract YourContractName # scaffold a contract
  $ truffle create test YourTestName         # scaffold a test

http://trufflesuite.com/docs
```

Slika 3. Inicijalizacija Truffle-a terminal



Slika 4. Inicijalizacija Truffle mape

Svaka mapa koja je kreirana ima svoju namjenu. U mapi pod nazivom contracts kreiramo datoteke tj. pametne ugovore koji su napisani u programskom jeziku Solidity. Nadalje mapa pod nazivom migrations služi za potrebe prelaska napisanog pametnog ugovora na određeni lanac blokova koji želimo ukoliko je kompatibilan. Ta migracijska skripta za prelazak biti će napisana u programskom jeziku JavaScript. Mapa test, kao što i sam naziv govori, služi za pisanje skripata sa kojima se vrše provjere na napravljeni pametni ugovor kako bi se izbjeglo postavljanje nepravilno napravljenog ugovora.



Slika 5. Kreiranje pametnog ugovora

Uz pomoć Truffle-a sve što je potrebno za kreiranje pametnog ugovora jest dodavanje nove datoteke u mapu contracts. Na slici 5. prikazana je i podcrtana crvenom bojom mapa sa nazivom SimpleStorage u kojoj se piše programski kod za željeni pametni ugovor.

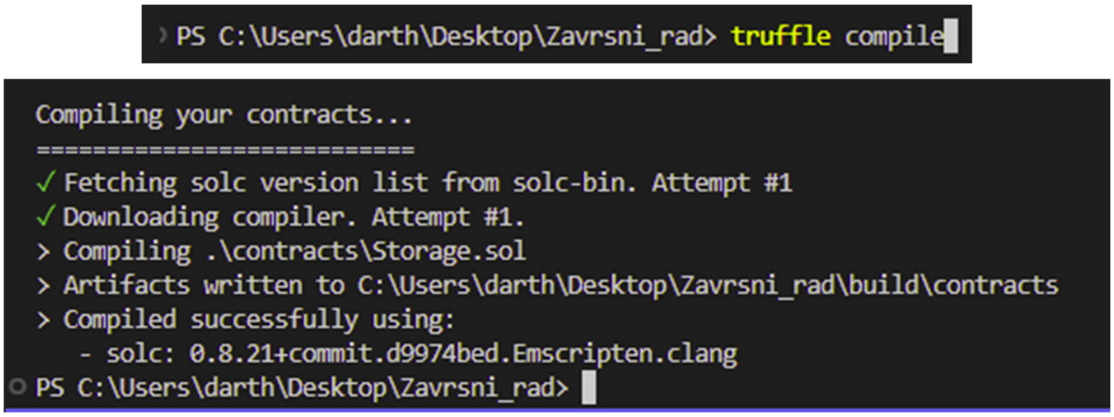
Slika 6. Programski kod pametnog ugovora

Ovo je osnovni primjer pametnog ugovora koji vam omogućuje da postavite i dobijete vrijednost jedne nepredznačene (unsigned) cjelobrojne varijable a. U nastavku će biti objašnjen programski kod sa slike 6. od kreiranog pametnog ugovora.

Objašnjenje koda:

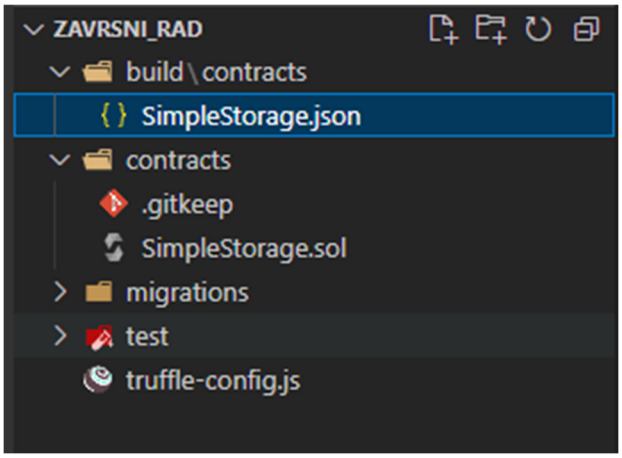
- // SPDX-License-Identifier: MIT: Ovo je komentar koji označava licencu pod kojom je ugovor objavljen. U ovom slučaju koristi MIT licencu.
- pragma solidity >=0.5.0 <0.9.0;: Ova izjava navodi verziju Solidityja s kojom je ugovor kompatibilan. To znači da se ugovor može sastaviti korištenjem Solidity verzija većih ili jednakih 0.5.0, ali manjih od 0.9.0

- contract SimpleStorage { ... }: Ovo je definicija ugovora. Sadrži sljedeće:
- uint256 a;: Ovo deklarira varijablu stanja a tipa uint256 (cijeli broj bez predznaka). Ova varijabla će pohraniti podatke koje možete postaviti i dobiti korištenjem dolje definiranih funkcija
- function setter(uint256 _a) public { ... }: Ovo je javna funkcija koja se zove postavljač. Potreban je jedan argument _a tipa uint256. Kada se ova funkcija pozove, ona postavlja vrijednost varijable a na vrijednost proslijeđenu kao _a. Ovu funkciju može pozvati bilo tko
- function getter() public view returns (uint256) { ... }: Ovo je javna funkcija koja se zove getter. Ne trebaju nikakvi argumenti. Označeno je kao prikaz, što znači da ne mijenja stanje ugovora. Jednostavno vraća trenutnu vrijednost varijable a kada se pozove. Ova je funkcija također dostupna svima i može se koristiti za čitanje vrijednosti a



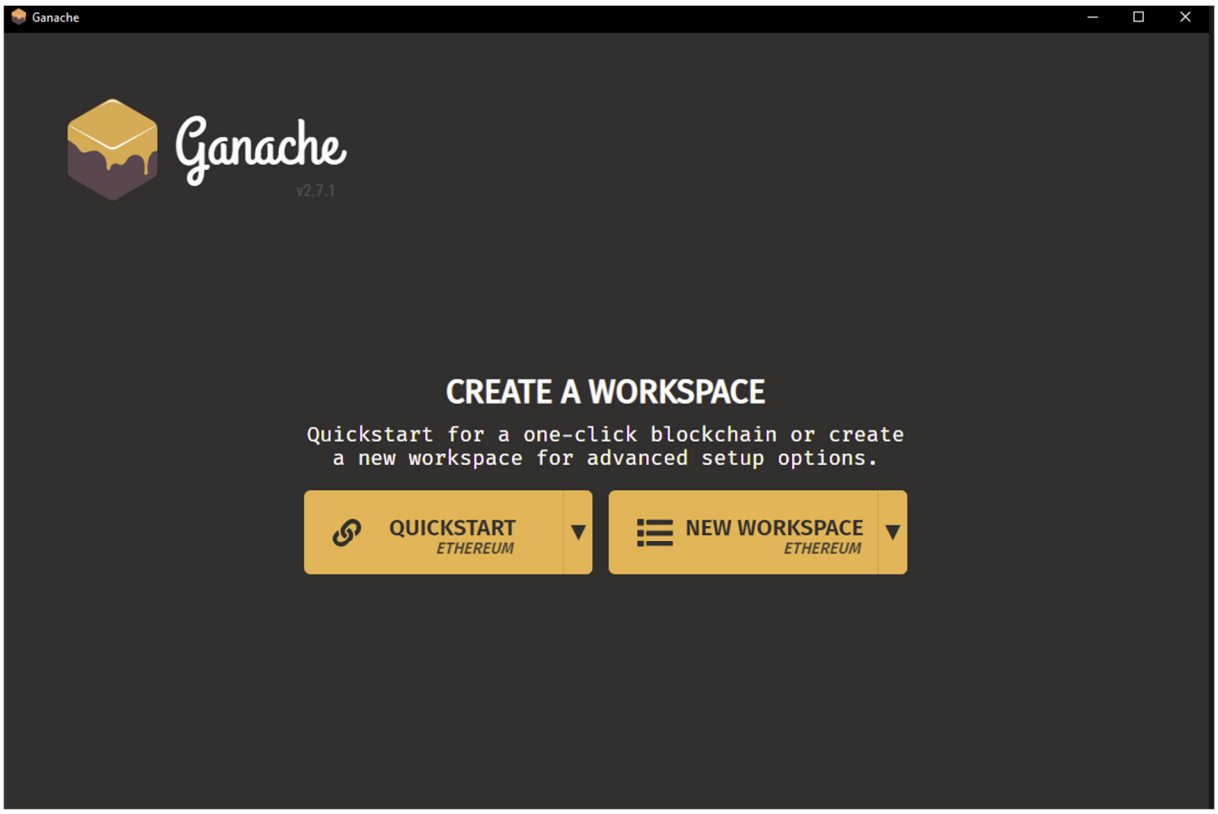
Slika 7. Kompilacija ugovora

Po završetku pisanja programskog koda za pametni ugovor nužno je uz pomoć komande truffle compile sastavititi pametni ugovor. Ukoliko je u terminalu nakon upisane komande prikazana poruka „Compiled succesfully using:“ kao što na slici 7., kompilacija je uspješno provedeno. Nakon kompilacije samostalno će se kreirat nova mapa pod nazivom „build“, slika 8., u kojoj se nalazi nova mapa contacts koja sadrži json datoteku istog imena kao što i sam ugovor, ali ekstenzije json umjesto sol. Prema Truffle-ovoj terminologiji kompilacijom pametnog ugovora kreiraju se „artefakti“. Artefakt je zapravo datoteka u kojoj se nalazi aplikacijsko binarno sučelje i bajt kod ugovora u ovom slučaju ugovora SimpleStorage.sol.



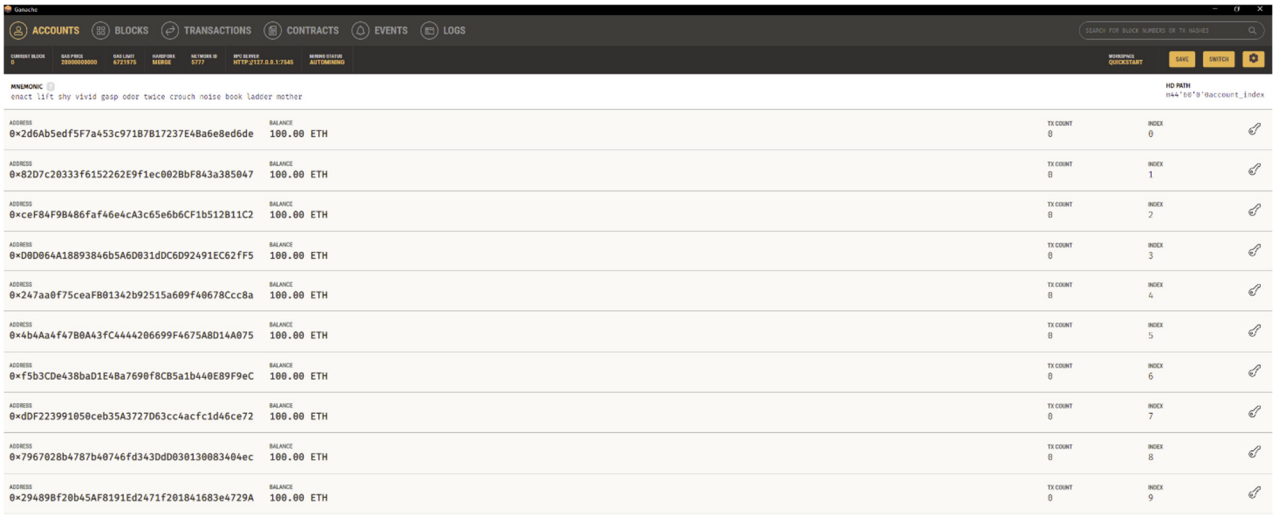
Slika 8. Generirana nova datoteka

Na temelju ovih primjera prikazano je i objašnjeno kako kreirat i sastaviti pametne ugovore. Naime, Truffle nudi mogućnost prebacivanja pametnih ugovora na razne lance blokova. Kako taj postupak zauzima prostor samog lanca bloka što prouzrokuje troškovima, za testiranje funkcionalnosti prebacivanja ugovora koristit će se Ganache.



Slika 9. Ganache

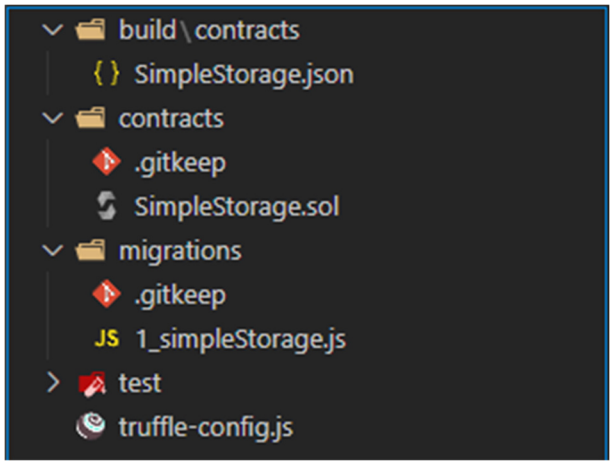
Nakon instalacije Ganache-a otvaranjem aplikacije dobivamo prikaz kao na slici 9. Odabirom prvog gumba pod nazivom „Quickstart“ kreira se privremeni lanac blokova.



Slika 10. Područje novčanika

Slika 11. Trenutni blokovi

Odabirom gumba „Quickstart“ otvara se područje aplikacije u kojemu se nalaze nasumično generirane adrese novčanika izmišljenih korisnika. Na slici 11. prikazana je alatna traka na kojoj su vidljiva sljedeća područja: accounts, blocks, transactions, contacts, events, logs. U području blocks prikazani je trenutno samo jedan blok koji se još naziva „genesis block“ kako je to prvi blok u ovome lancu blokova. Korištena područja u svrhu ovoga rada biti će područje accounts, blocks i transactions.



Pomoću Truffle-a kreiran i kompiliran je pametan ugovor. Upotrebom Ganache-a uspostavljen je lanac blokova na koji se može prebacit pametan ugovor. Za migraciju ugovora nužno je kreiranje skripte koja će to učiniti. Prvo što je potrebno učiniti jest kreirati novu datoteku sa ekstenzijom js u mapi migrations. Nadalje je nužno napisat kod za implementaciju postavljanja pametnoga ugovora na lanac blokova.

Slika 12. Implementacija skripte postavljanja

Slika 12. prikazuje kod korišten za implementaciju funkcionalnosti postavljanja. Ovaj kod je JavaScript skripta koja se koristi u kontekstu razvoja Ethereum pametnih ugovora pomoću Truffle okvira.

Objašnjenje koda jest sljedeće:

- var SimpleStorage = artifacts.require("./SimpleStorage.sol");: U ovoj liniji uvozi se artefakt pametnog ugovora pomoću Truffle-ovog sustava artefakata. Varijabli SimpleStorage dodjeljuje se objekt ugovora pametnog ugovora definiranog u datoteci

"SimpleStorage.sol". Ovi artefakti generiraju se prilikom kompilacije pametnih ugovora pomoću Truffle-a i pružaju način za programsko interagiranje i implementaciju ugovora

- module.exports = function (deployer) { ... }: Ovo je uobičajeni uzorak u Node.js za izvoz funkcija ili objekata kako bi ih se učinilo dostupnima u drugim dijelovima koda ili modulima. U ovom slučaju izvozi se funkcija koja prima objekt deployer kao argument

- deployer.deploy(SimpleStorage);: Unutar izvožene funkcije koristi se objekt deployer kako bi se specificirala želja za implementirati pametni ugovor SimpleStorage. Ova linija govori Truffle-u da implementira navedeni ugovor na Ethereum mreži koja se konfigurirala u Truffle projektu

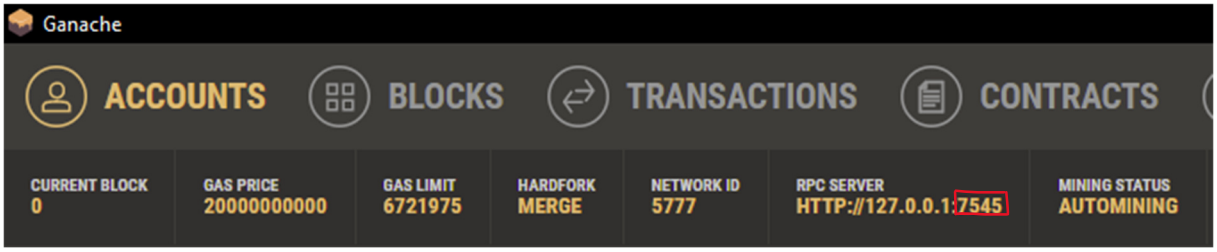
Ukratko, ovaj kod uvozi artefakt pametnog ugovora SimpleStorage i izvozi funkciju koja govori Truffle-u da implementira ovaj ugovor kada se pokrene postavljanje. Ovo je tipičan obrazac u Truffle razvoju gdje se definiraju koraci implementacije pametnih ugovora pomoću JavaScript skripti.

Sada kada je kreirana i skripta postavljanja nedostaje 1 korak za uspješno prebacivanje izrađenog pametnog ugovora na lanac blokova. Taj korak jest promjena truffle-config.js datoteke kako bi sam Truffle znao na koji lanac blokova prebacit pametan ugovor.

```
//  development: {
//  host: "127.0.0.1",     // Localhost (default: none)
//  port: 8545,            // Standard Ethereum port (default: none)
//  network_id: "*",       // Any network (default: none)
// },
```

Slika 13. Development dio truffle-config.js-a

Potrebno je pronaći development sekciju u truffle-config.js-u. Sekcija o kojoj je riječ je prikazana na slici 13. Prvo je potrebno osloboditi sekciju što znači izbrisat da je komentirana. Nadalje potrebno je provjeriti port za server kod Ganache-a, na slici 14. unutar crveno označenog područja. Kada je to sve učinjeno promjene bi trebale izgledati kao na slici 15.

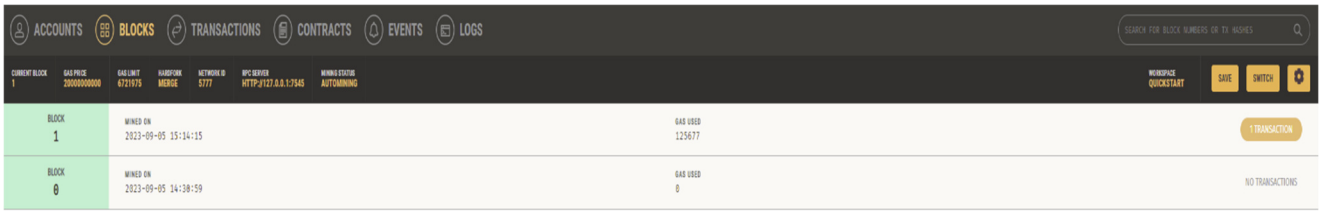


Slika 14.Ganache RPC server

```
//
  development: {
  host: "127.0.0.1",     // Localhost (default: none)
  port: 7545,            // Standard Ethereum port (default: none)
  network_id: "*",       // Any network (default: none)
  },
//
```

Slika 15.Development dio truffle-config.js-a (promjena)

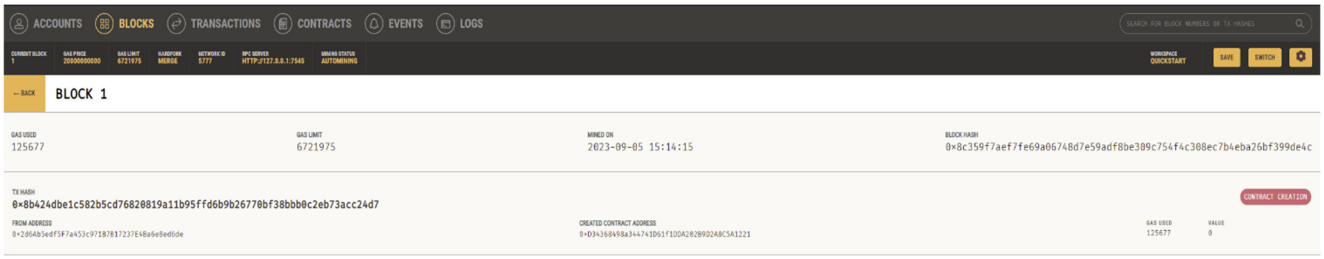
Ako su svi koraci ispunjeni kao na slikama jedna komanda nas udaljava od uspješnog postavljanja kreiranog pametnog ugovora na lanac blokova. Komanda o kojoj je riječ jest truffle migrate. Ukoliko je sve ispravno učinjeno prikaz bi trebao biti kao na slici 18. Važno je napomenuti da kako nije postavljena adresa novčanika koji zapravo postavlja pametan ugovor na lanac blokova, uobičajeno je da će odabrana adresa novčanika biti prva koja se nalazi u privremenom lancu blokova koji je kreiran uz pomoć Ganache-a. Na slici 16. vidljivo je da je dodan jedan novi blok u ovaj lanac kako je uspješno postavljen pametan ugovor.



Slika 16. Prikaz blokova

Iz slike 17. koja je prikaz novokreiranog bloka 1, u lijevom kutu vidljiva je adresa novčanika koja je postavila pametan ugovor na lanac blokova. Ta adresa je adresa prvog novčanika iz područja „Accounts“. Nadalje na slici 18. vidljive su stavke kao balance, gas used i mnoge druge. Kako dodavanje bilo kakvog novog bloka, u ovom slučaju postavljanje pametnog ugovora na lanac blokova, zauzima slobodan prostor lanca blokova taj prostor bit će naplaćen.

Iz tog razloga je oduzeta vrijednost od balance-a adrese novčanika. Početan balance bio je 100 ETH, a sada je smanjen za vrijednost „Total cost“ sa slike 18.



Slika 17. Prikaz bloka 1

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL

Starting migrations...
======================
> Network name:    'development'
> Network id:      5777
> Block gas limit: 6721975 (0x6691b7)


1_simpleStorage.js
==================

   Deploying 'SimpleStorage'
   -------------------------
   > transaction hash:    0x8b424dbe1c582b5cd76820819a11b95ffd6b9b26770bf38bbb0c2eb73acc24d7
   > Blocks: 0            Seconds: 0
   > contract address:    0xD34368498a344741D61f1DDA202B9D2A8C5A1221
   > block number:        1
   > block timestamp:     1693919655
   > account:             0x2d6Ab5edf5F7a453c971B7B17237E4Ba6e8ed6de
   > balance:             99.999575840125
   > gas used:            125677 (0x1eaed)
   > gas price:           3.375 gwei
   > value sent:          0 ETH
   > total cost:          0.000424159875 ETH

   > Saving artifacts
   -------------------------------------
   > Total cost:      0.000424159875 ETH

Summary
=======
> Total deployments:   1
> Final cost:          0.000424159875 ETH
```

Slika 18. Prikaz postavljanja na lanac blokova

4.4.1. Kako koristiti prebačeni pametni ugovor?

Nakon što se pametni ugovor postavi na lanac blokova, ljudi mogu komunicirati s njim slanjem transakcija na adresu ugovora. Korisnici moraju znati adresu pametnog ugovora koji je implementiran. Ova se adresa obično daje kada se ugovor implementira i dostupna je putem preglednika lanca blokova ili aplikacije koja je implementirala ugovor. Za interakciju s pametnim ugovorom korisnici obično trebaju kompatibilni novčanik lanca blokova. Ovaj se novčanik koristi za potpisivanje i slanje transakcija. Popularni novčanici uključuju MetaMask,

Trust Wallet i hardverske novčanike kao što su Ledger i Trezor. Korisnici moraju povezati svoj novčanik s mrežom lanca blokova na kojoj je implementiran pametni ugovor. Decentralizirane aplikacije ili web sučelja obično se koriste za pristup pametnim ugovorima. Korisnici povezuju svoj novčanik s pametnim ugovorom posjetom decentralizirane aplikacije ili web stranici koja s njim komunicira. Ovisno o dizajnu pametnog ugovora, korisnici mogu pozvati više funkcija ili metoda. Ove funkcije predstavljaju radnje koje ugovor može izvršiti. Korisnici biraju koju funkciju žele koristiti. Korisnici unose te vrijednosti u aplikaciji ili sučelje ako odabrana funkcija zahtijeva ulazne parametre (kao što je iznos prijenosa ili podaci). Nakon postavljanja parametara, korisnici potvrđuju transakciju unutar svog novčanika. Ova radnja generira digitalni potpis koji dokazuje vlasništvo nad novčanikom i autorizira transakciju. Nakon toga se potpisana transakcija predaje mreži lanca blokova. Ako transakcija zadovoljava potrebne kriterije, rudari ili validatori na mreži će je provjeriti i uključiti u blok. Ovisno o brzini lanca blokova i zagušenju, može proći neko vrijeme dok se transakcija ne potvrdi. Korisnici mogu pratiti status svoje transakcije na pregledniku lanca blokova. Nakon što je transakcija potvrđena i dodana u blok, pametni ugovor izvršava odabranu funkciju na temelju navedenih parametara. Korisnici mogu pregledati rezultate transakcije, koji mogu uključivati promjene stanja pametnog ugovora, prijenose imovine ili druge radnje navedene kodom ugovora. Korisnici mogu nastaviti komunicirati s pametnim ugovorom pozivanjem drugih funkcija ili izvođenjem dodatnih radnji prema potrebi. Važno je napomenuti da se specifični koraci i korisničko iskustvo mogu razlikovati ovisno o platformi lanca blokova koja se koristi, dizajnu pametnog ugovora i korisničkom sučelju koje pruža decentralizirana aplikacija ili web mjesto. Nadalje, korisnici moraju biti oprezni pri interakciji s pametnim ugovorima jer su transakcije na lancu blokova obično nepovratne, a netočni unosi mogu rezultirati gubitkom imovine.

5. Zaključak

Ukratko, tehnologija lanca blokova je transformativna sila u našem digitalnom svijetu. Njegova decentralizirana, transparentna i nepromjenjiva priroda ima potencijal transformirati industrije, poboljšati sigurnost i redefinirati naše interakcije s razmjenom podataka i vrijednosti. Dok nastavljamo otključavati ogroman potencijal lanca blokova, postaje jasno da je ova inovativna tehnologija više od samo prolaznog hira, već temeljni stup digitalnog doba. Prihvaćanje potencijala lanca blokova uz rješavanje njegovih izazova bit će ključno dok se krećemo u dinamičnoj budućnosti koju obećava. Zaključno, okviri lanca blokova pojavili su se kao transformativni alati s dalekosežnim implikacijama u raznim industrijama. Dok nastavljamo istraživati i razvijati te okvire, očito je da je tehnologija lanca blokova spremna preoblikovati način na koji razmjenjujemo vrijednost i informacije, uvodeći novu eru inovacija i poremećaja u digitalnom dobu. Prihvaćanje i prilagodba okvirima lanca blokova bit će ključno za organizacije i pojedince kako bi napredovali u ovom krajoliku koji se brzo razvija. Iz praktičnog dijela vidljivo je kako korištenje okvira olakšava rad sa razvijanjem tehnologije lanca blokova. Svaki okvir ima posebnu namjenu tako i okvir Truffle koji je namijenjen za olakšano razvijanje, sastavljanje i prenašanje pametnih ugovora na lance blokova. Truffle kao takav je jedan od mnogih okvira koji također potiču potencijalne korisnike i razvojne programere da na jednostavan način pristupe u svijet lanca blokova i njegov razvoj.

Popis literature

[1] „What is blockchain? | McKinsey“. https://www.mckinsey.com/featured-insights/mckinsey-explainers/what-is-blockchain (pristupljeno 24. lipanj 2023.).

[2] „What is Decentralization?“, *Amazon Web Services, Inc.* https://aws.amazon.com/blockchain/decentralization-in-blockchain/ (pristupljeno 26. lipanj 2023.).

[3] „Advantages and Disadvantages of Decentralized Blockchains“, *World Crypto Index*. https://www.worldcryptoindex.com/advantages-disadvantages-decentralized-blockchains/ (pristupljeno 26. lipanj 2023.).

[4] „Distributed Ledger Technology (DLT): Definition and How It Works“, *Investopedia*. https://www.investopedia.com/terms/d/distributed-ledger-technology-dlt.asp (pristupljeno 26. lipanj 2023.).

[5] „What is Cryptography? Definition, Importance, Types“, *Fortinet*. https://www.fortinet.com/resources/cyberglossary/what-is-cryptography (pristupljeno 26. lipanj 2023.).

[6] „What Are Consensus Mechanisms in Blockchain and Cryptocurrency?“, *Investopedia*. https://www.investopedia.com/terms/c/consensus-mechanism-cryptocurrency.asp (pristupljeno 26. lipanj 2023.).

[7] „What are smart contracts on blockchain? | IBM“. https://www.ibm.com/topics/smart-contracts (pristupljeno 26. lipanj 2023.).

[8] „What Are Permissioned and Permissionless Blockchains?“, *Binance Academy*. https://academy.binance.com/en/articles/what-are-permissioned-and-permissionless-blockchains (pristupljeno 18. srpanj 2023.).

[9] R. MacDonald, „Permissionless vs. Permissioned Blockchains: Pros & Cons“, *1Kosmos*, 14. srpanj 2022. https://www.1kosmos.com/blockchain/permissionless/ (pristupljeno 18. srpanj 2023.).

[10] „What is Tokenization?“, *CoinGeek*. https://coingeek.com/bitcoin101/what-is-tokenization/ (pristupljeno 18. srpanj 2023.).

[11] Crypto.com, „A Deep Dive Into Blockchain Scalability“. https://crypto.com/university/blockchain-scalability (pristupljeno 13. srpanj 2023.).

[12] „What Is A Cross Chain Bridge? | Chainlink“. https://chain.link/education-hub/cross-chain-bridge (pristupljeno 19. srpanj 2023.).

[13] „Bitcoin Atomic Swaps: What, How They Work | Trust Machines“. https://trustmachines.co/learn/bitcoin-atomic-swaps-how-they-work/ (pristupljeno 19. srpanj 2023.).

[14] „An Introduction to Sidechains“, 07. ožujak 2022. https://www.coindesk.com/learn/an-introduction-to-sidechains/ (pristupljeno 19. srpanj 2023.).

[15] „What Is Blockchain Interoperability? | Chainlink“. https://chain.link/education-hub/blockchain-interoperability (pristupljeno 19. srpanj 2023.).

[16] „What Is an Oracle in Blockchain? » Explained | Chainlink“. https://chain.link/education/blockchain-oracles (pristupljeno 20. srpanj 2023.).

[17] „Hashed Timelock Contract (HTLC): Overview and Examplesin Crypto“, *Investopedia*. https://www.investopedia.com/terms/h/hashed-timelock-contract.asp (pristupljeno 20. srpanj 2023.).

[18] „What Are Public Keys Vs Private Keys?“, *Ledger*. https://www.ledger.com/academy/blockchain/what-are-public-keys-and-private-keys (pristupljeno 19. srpanj 2023.).

[19] „Hards Forks and Soft Forks Explained“, *Binance Academy*. https://academy.binance.com/en/articles/hard-forks-and-soft-forks (pristupljeno 19. srpanj 2023.).

[20] „What Is a Framework?“ https://www.codecademy.com/resources/blog/what-is-a-framework/ (pristupljeno 23. lipanj 2023.).

[21] M. Blog, „Hardhat Explained – What is Hardhat?“, *Moralis Web3 | Enterprise-Grade Web3 APIs*, 24. lipanj 2021. https://moralis.io/hardhat-explained-what-is-hardhat/ (pristupljeno 25. srpanj 2023.).

[22] „Introduction to Foundry | QuickNode“, 18. kolovoz 2023. https://www.quicknode.com/guides/ethereum-development/smart-contracts/intro-to-foundry (pristupljeno 30. kolovoz 2023.).

[23] „Tatum - Framework“. https://tatum.io/framework (pristupljeno 31. kolovoz 2023.).

[24] „About - Curvegrid“. https://www.curvegrid.com/about (pristupljeno 31. kolovoz 2023.).

[25] „ApeWorx, LTD - The WEB3 Tool From the Future“. https://www.apeworx.io/ (pristupljeno 31. kolovoz 2023.).

[26] „Robot Framework Solidity Testing Toolkit - Web3 Testing Tools - Alchemy“. https://www.alchemy.com/dapps/robot-framework (pristupljeno 05. rujan 2023.).

[27] „Command palette - Truffle Suite“. https://trufflesuite.com/docs/vscode-ext/reference/command-palette/ (pristupljeno 05. rujan 2023.).

[28] D. Drescher, *Blockchain Basics: A Non-Technical Introduction in 25 Steps*. Frankfurt am Main, Germany: Apress.

[29] H. Diedrich, *Ethereum: Blockchains, Digital Assets, Smart Contracts, Decentralized Autonomous Organizations*. Wildfire Publishing, 2016.

Popis slika

Popis tablica