

SVEUČILIŠTE U ZAGREBU  
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 148

**NEZAMJENJIVI TOKENI TEMELJENI NA CARDANO LANCU  
BLOKOVA**

Sven Skender

Zagreb, lipanj 2023.

SVEUČILIŠTE U ZAGREBU  
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 148

**NEZAMJENJIVI TOKENI TEMELJENI NA CARDANO LANCU  
BLOKOVA**

Sven Skender

Zagreb, lipanj 2023.

Zagreb, 10. ožujka 2023.

## **DIPLOMSKI ZADATAK br. 148**

Pristupnik: **Sven Skender (0036507654)**

Studij: Računarstvo

Profil: Znanost o podacima

Mentor: doc. dr. sc. Marko Horvat

Zadatak: **Nezamjenjivi tokeni temeljeni na Cardano lancu blokova**

Opis zadatka:

Cardano lanac blokova može se koristiti za skalabilnu i učinkovitu izradu pametnih ugovora, decentraliziranih aplikacija i protokola, uključujući nezamjenjive tokene (NFT) kao jedinstvene javno dijeljene identifikatore digitalnih dokumenata. Instalirati i pokrenuti Cardano čvor. Razmotriti i objasniti sigurnosne aspekte i tehničke koncepte koji uključuju transakcije, adrese i validaciju blokova. Razviti programsku podršku koja omogućuje kreiranje novih NFT tokena i njihovu pohranu na Cardano lanac blokova s dodatnim metapodacima. Sadržaj tokena obavezno uključuje digitalnu sliku i geolokaciju. Implementirati pohranu digitalnih dokumenata u IPFS "peer-to-peer" distribuirani podatkovni sustav i lanac blokova. Omogućiti dohvaćanje podataka iz realiziranog lanca blokova te prikazati njihov sadržaj i metapodatake. U implementaciji koristiti programski jezik JavaScript i okvir Node.js. U diplomskom radu opisati Cardano platformu, korištenu razvojnu okolinu i druga programska sredstva te postupke potrebne za izradu aplikacije. Prikazati upute za instalaciju razvijenog sustava. U diplomskom radu prikazati opise obrazaca uporabe, isječke izvornog programskog koda i njihove grafičke prikaze rada uz potrebna dodatna objašnjenja i dokumentaciju. Citirati korištenu literaturu.

Rok za predaju rada: 23. lipnja 2023.

*Duboko sam zahvalan svom mentoru, čije su znanje i iskustvo oblikovali moj rast i razvoj tijekom naše suradnje. Također bih želio izraziti svoju iskrenu zahvalnost svojoj zaručnici, čija je vjera u mene i podrška bili beskrajni izvor snage i ohrabrenja. Mojoj obitelji, koja je uvijek postavljala pitanje "Jesi li položio?" nakon samo dva tjedna od početka akademske godine, duboko sam zahvalan na kontinuiranoj motivaciji i vjeri u moje sposobnosti. Na kraju, želim izraziti svoju iskrenu zahvalnost svima koji su bili uz mene i pružili podršku kada mi je bila najpotrebnija.*

# SADRŽAJ

<b>1. Uvod</b>	<b>1</b>
<b>2. Koncept tokenizacije na lancu blokova</b>	<b>2</b>
2.1. Opis lanca blokova . . . . .	2
2.2. Svojstvo zamjenjivosti unutar tokenizacije . . . . .	4
2.2.1. Zamjenjivi tokeni i primjena . . . . .	4
2.2.2. Nezamjenjivi tokeni i primjena . . . . .	5
<b>3. Aplikacija za nezamjenjive tokene (ANT)</b>	<b>6</b>
3.1. Poslužiteljska aplikacija . . . . .	6
3.1.1. Izvršno okruženje . . . . .	7
3.1.2. Razvojni okvir . . . . .	8
3.2. Cardano lanac blokova . . . . .	9
3.2.1. Mehanizam konsenzusa . . . . .	9
3.2.2. Model nepotrošenog izlaza transakcije . . . . .	10
3.2.3. Izvorni tokeni . . . . .	12
3.2.4. Politika kovanja tokena . . . . .	12
3.3. Kontejnerizacija programskog rješenja . . . . .	14
3.3.1. Virtualizacija i kontejnerizacija . . . . .	14
3.3.2. Docker i mikroservisi . . . . .	16
3.3.3. Komunikacija između procesa . . . . .	17
<b>4. Implementacija programskog rješenja</b>	<b>20</b>
4.1. Arhitektura programskog rješenja . . . . .	20
4.2. Pokretanje Cardano čvora i sinkronizacija . . . . .	21
4.3. Instalacija aplikacije . . . . .	24
4.4. Postavke aplikacije i kripto novčanika . . . . .	28
4.5. Status i prijenos sredstava . . . . .	30

4.6. Kovanje nezamjenjivih tokena . . . . .	34
4.7. Pregled rezultata . . . . .	43
<b>5. Zaključak</b>	<b>45</b>
<b>Literatura</b>	<b>46</b>

# 1. Uvod

Lanac blokova (engl. *blockchain*) je temeljna tehnologija proizašla iz područja digitalnih valuta. Izvorno je bila prisutna samo u svijetu financijskih transakcija te pokretala kripto valutu Bitcoin, ali je promijenila svoj doseg i utjecala na različite sektore, mijenjajući način na koji se informacije i transakcije bilježe i provjeravaju. Tradicionalni sustavi vođenja zapisa i obrade transakcija dugo su se oslanjali na centralizirane posrednike kao što su banke, vlade i institucije treće strane. Međutim, ove centralizirane strukture podložne su ljudskim pogreškama, manipulacijama i zlonamjernim napadima. Lanac blokova, s druge strane, uvodi promjenu paradigme omogućujući distribuiranoj mreži sudionika da postignu konsenzus i održavaju zajedničku knjigu bez potrebe za posrednicima.

Na svojoj najosnovnijoj razini, lanac blokova čini nepromjenjivi i sekvencijalni lanac blokova, od kojih svaki sadrži niz provjerenih i vremenski označenih transakcija. Ovaj se lanac kontinuirano ažurira i replicira na više računala ili čvorova unutar mreže. Decentralizirana priroda sustava osigurava da niti jedan entitet nema potpunu kontrolu nad cijelim sustavom, čime se smanjuje rizik od manipulacije i povećava transparentnost. Implikacije lanca blokova protežu se daleko izvan područja financija, obuhvaćajući različite sektore, uključujući upravljanje opskrbnim lancem, zdravstvo, državne službe i prava intelektualnog vlasništva.

Jedna od mogućih primjena nalazi se i u humanitarnom razminiranju. Humanitarno razminiranje igra ključnu ulogu u postkonfliktnoj obnovi jer prisutnost mina i neeksplodiranih sredstava predstavlja ozbiljnu prijetnju civilima i infrastrukturi. Proces protuminskog djelovanja uključuje prikupljanje podataka iz različitih izvora, uključujući daljinska istraživanja, stručno znanje o protuminskom djelovanju, formalne i neformalne povijesne zapise, kao i intervju sa stanovništvom, bivšim borcima i očevicima. Kako nove informacije postaju dostupne ili se situacija na terenu razvija, nalazi istraživanja zahtijevaju periodična ažuriranja, a tako se pojavljuje i problem pouzdanosti podataka i skladištenja. Lanac blokova se tako pokazuje kao odličan sustav koji bi mogao trajno riješiti problem pouzdanosti podataka i nepovjerenja među više dionika.

## 2. Koncept tokenizacije na lancu blokova

Kada je riječ o pohrani podataka o minama i neeksplozivnim ubojitim sredstvima, lokacijama i dodatnim metapodacima, lanac blokova pruža robusno rješenje. Tokenizacijom ovih informacija na lancu, svako eksplozivno sredstvo može se predstaviti kao jedinstvena digitalna imovina, omogućujući precizno praćenje i upravljanje. Decentralizirana priroda lanca blokova osigurava da se podaci distribuiraju na više čvorova, što ih čini otpornim na neovlašteno mijenjanje pa se time onemogućava skrivanje i brisanje podataka. Ova transparentnost i nepromjenjivost donose značajne prednosti, uključujući povećanu odgovornost, sljedivost i povjerenje u podatke. Uz lanac blokova, dionici mogu pouzdano pristupiti i provjeriti informacije povezane s minama, smanjujući rizik od slučajnih eksplozija, pomažući u naporima razminiranja i poboljšavajući opće sigurnosne standarde.

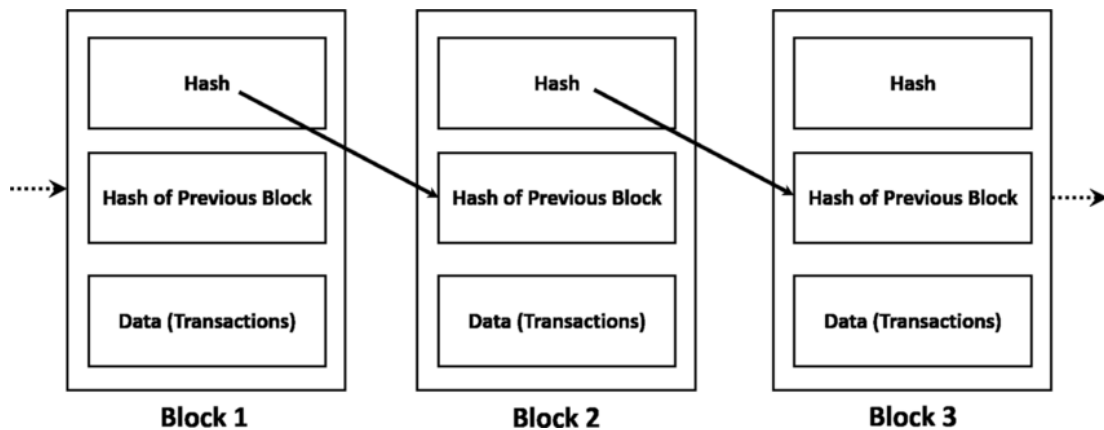
### 2.1. Opis lanca blokova

Lanac blokova je revolucionarna tehnologija koja iz temelja mijenja način na koji se podaci pohranjuju i upravljaju njima [5]. U svojoj srži, lanac blokova je decentralizirana i distribuirana digitalna knjiga koja bilježi transakcije ili informacije na više računala, poznatih kao čvorovi [6]. Ova decentralizirana priroda eliminira potrebu za središnjim tijelom, poput banke ili vlade, za provjeru i autentifikaciju transakcija. Umjesto toga, povjerenje se uspostavlja pomoću algoritama konsenzusa i kriptografskih tehnika. U lancu blokova, svaka transakcija ili dio informacije grupirani su u blok. Ti se blokovi zatim međusobno povezuju u kronološki i nepromjenjivi lanac, tvoreći potpuni lanac blokova [20]. Svaki blok ima svoj sažetak (engl. *hash*) koji proizlazi iz svih informacija pohranjenih u tom bloku te pokazuje na prethodni blok kako je vidljivo na dijagramu 3.2. Podaci pohranjeni unutar blokova zaštićeni su pomoću naprednih kriptografskih algoritama, što čini ekstremno teškim za bilo koga da promijeni



informacije ili same blokove, a da ne bude otkriven [12].

Jedna od ključnih značajki ove tehnologije je njezina transparentnost. Svaki sudionik u mreži ima pristup kopiji cijelog lanca, čime se osigurava da su sve transakcije i promjene podataka vidljive svima. Ova transparentnost promiče povjerenje i odgovornost, budući da se svaka zlonamjerna aktivnost ili pokušaj manipulacije može lako prepoznati i odbiti od strane mreže [11]. Tehnologija omogućuje i poboljšanu sigurnost kroz mehanizme konsenzusa. Ovi mehanizmi osiguravaju da se svi čvorovi u mreži slažu oko valjanosti transakcija prije nego što se dodaju u lanac [24]. Taj se dogovor postiže različitim algoritmima konsenzusa, kao što su dokaz o radu (engl. *proof of work*) ili dokaz o udjelu (engl. *proof of stake*), koji zahtijevaju od sudionika da osiguraju računalnu snagu ili ulože vlastitu kriptovalutu (engl. *cryptocurrency*) za provjeru valjanosti transakcija.



Slika 2.1: Pojednostavljeni dijagram lanca blokova [9]

Iskorištavanjem ovih temeljnih načela, ova tehnologija može se primijeniti na različite industrije i slučajeve upotrebe, uključujući upravljanje podacima o minama i neeksplodiranim sredstvima. Osigurava integritet i pouzdanost podataka sigurnim pohranjivanjem relevantnih informacija u nepromjenjivu i transparentnu glavnu knjigu. Decentralizirana priroda lanca blokova olakšava suradnju među dionicima, što dovodi do poboljšane raspodjele resursa, smanjenog dupliciranja napora i boljeg donošenja odluka u operacijama razminiranja. Štoviše, mogućnost provjere koju pruža lanac blokova povećava povjerenje u informacije i omogućuje prepoznavanje potencijalnih nedosljednosti u procesu otkrivanja mina.

## 2.2. Svojstvo zamjenjivosti unutar tokenizacije

Tokenizacija je proces pretvaranja imovine iz stvarnog svijeta ili digitalne imovine u tokene koji se mogu pohranjivati i prenositi na lancu blokova [13]. Tokeni, u kontekstu lanca blokova, digitalni su prikazi imovine ili prava. Mogu se stvarati, posjedovati i prenositi, pružajući sigurnu i transparentnu metodu predstavljanja vrijednosti. Tokeni mogu predstavljati širok raspon imovine, uključujući fizičku imovinu poput nekretnina i robe, kao i nematerijalnu imovinu poput intelektualnog vlasništva, bodova vjernosti, prava pristupa određenim uslugama, sudjelovanju u procesima upravljanja itd. Vrijednost tokena obično se izvodi iz temeljne imovine ili prava koja predstavlja [13].

Kripto valuta je, s druge strane, specifična vrsta digitalne imovine koja služi kao sredstvo razmjene, baš kao i tradicionalne valute poput američkog dolara ili eura [24]. Glavna razlika između tokena i kripto valuta leži u njihovoj funkcionalnosti i namjeni. Dok su kripto valute poput Bitcoina <sup>1</sup>, Ethera <sup>2</sup> i Ade <sup>3</sup> osmišljene primarno kao mediji razmjene ili pohrane vrijednosti, tokeni mogu predstavljati mnogo širi raspon imovine i imaju specifične slučajeve upotrebe unutar ekosustava lanca blokova. Kripto valute funkcioniraju kao samostalne digitalne valute koje se mogu koristiti za razne digitalne transakcije te često rade na vlastitim lancima blokova te imaju vlastita pravila i protokole. S druge strane, tokeni se oslanjaju na postojeće platforme, kao što su Ethereum ili Cardano, za stvaranje i njihov rad. Kako se obično grade na temeljima postojećih infrastrukture, nasljeđuju njihovu sigurnost, skalabilnost i mehanizme konsenzusa. Tokeni tako nude veću fleksibilnost i prilagodbu u usporedbi s kripto valutama.

### 2.2.1. Zamjenjivi tokeni i primjena

Zamjenjivi tokeni (engl. *fungible token*) su vrsta tokena koji su međusobno zamjenjivi i identični. Svaka jedinica zamjenjivog tokena ima istu vrijednost kao i bilo koja druga jedinica te se na tom principu mogu razmjenjivati [8]. Klasičan primjer zamjenjivih tokena su kripto valute poput Bitcoina, Ethera ili Ade. Na primjer, ako se jedan Bitcoin zamijeni za drugi, vrijednost ostaje ista.

Zamjenjivi tokeni nalaze široku primjenu u raznim aplikacijama za decentralizirano financiranje. U aplikacijama za decentralizirano financiranje (engl. *decentralized finance*, *DeFi*) tokeni koji su vezani za vrijednost fiksne valute poput američkog dolara, pružaju stabilnost i obično se koriste za trgovanje [21]. Kao što je u stvarnom svijetu

---

<sup>1</sup><https://bitcoin.org/en/>

<sup>2</sup><https://ethereum.org/en/>

<sup>3</sup><https://cardano.org/>

moguće zamijeniti bilo koju novčanicu jednog američkog dolara za bilo koju drugu, tako je i s tokenima koji predstavljaju vrijednost fiksne valute. Na taj način korisnici mogu trgovati tokeniziranim američkim dolarima na lancu blokova. Decentralizirani financijski protokoli tako koriste zamjenjive tokene za omogućavanje pozajmljivanja, uzgoja prinosa i drugih financijskih usluga, gdje korisnici mogu osigurati likvidnost i zaraditi kamate.

### **2.2.2. Nezamjenjivi tokeni i primjena**

Nasuprot tome, nezamjenjivi tokeni (engl. *non-fungible token*, *NFT*) predstavljaju jedinstvenu digitalnu imovinu koja je nedjeljiva i razlikuje se jedna od druge [22]. Svaki nezamjenjivi token posjeduje određeni identifikator koji ga izdvaja i daje mu inherentnu vrijednost. Za razliku od zamjenjivih tokena, nezamjenjivi tokeni se ne mogu izravno razmjenjivati na bazi jedan-na-jedan zbog svojih jedinstvenih karakteristika. Nezamjenjivi tokeni su posljednjih godina privukli značajnu pozornost, prvenstveno u domenama digitalne umjetnosti, kolekcionarstva i igara [3].

Nezamjenjivi tokeni su revolucionirali svijet umjetnosti omogućivši umjetnicima da tokeniziraju svoje kreacije i prodaju ih izravno kolekcionarima [3]. Umjetnici mogu priložiti meta podatke svojim nezamjenjivi tokenima, pružajući dokaz o vlasništvu, autentičnosti i porijeklu. Ovo je otvorilo nove puteve za umjetnike da dopru do globalne publike, dobiju poštenu naknadu za svoj rad i zadrže kontrolu nad svojim intelektualnim vlasništvom. Pojavila su se tržišta nezamjenjivi tokena na kojima kolekcionari mogu kupovati, prodavati i trgovati digitalnim umjetninama, stvarajući svoj ekosustav za digitalne umjetnosti [4]. Štoviše, nezamjenjivi tokeni su pronašli korist i u industriji igara. Mogu predstavljati predmete u igri, likove ili virtualnu zemlju. Igrači mogu posjedovati ovu digitalnu imovinu, trgovati njome s drugim igračima, pa čak i koristiti je na više platformi za igranje.

Osim umjetnosti i igara, nezamjenjivi tokeni imaju potencijalne primjene i u područjima kao što su izdavanje ulaznica, virtualne nekretnine, prava intelektualnog vlasništva, upravljanje opskrbnim lancem itd. K tome, mogu poslužiti kao odlično sredstvo za pohranjivanje jedinstvenih geoprostornih informacija na kojima se nalaze točke od interesa u javnu i decentraliziranu bazu podataka. Tako je moguće pohranjivati lokacije raznih kampova, izvora pitke vode, SOS telefona, nalazišta rijetkih ptica, eksplozivnih i drugih opasnih naprava u bazu podataka koja ima ugrađen mehanizam za dokazivanje vlasništva, autentičnosti i jedinstvenosti.

## **3. Aplikacija za nezamjenjive tokene (ANT)**

U sklopu ovog rada razvijena je aplikacija za kreiranje nezamjenjivih tokena na lancu blokova (ANT). Aplikacija služi kao posrednik između raznih aplikacija koje se mogu koristiti u procesu humanitarnoga razminiranja za obradu i skladištenje podataka te samog lanca blokova koji omogućava sigurnost, transparentnost i povjerljivost. Prikaz arhitekture, upute za instalaciju i korištenje sustava bit će detaljno opisane u sljedećem poglavlju. U ovom poglavlju bit će opisano istraživanje tehnologija korištenih za razvoj i implementaciju samog programskog rješenja. Bit će navedeni i objašnjeni razni alati, razvojni okviri i platforme koje će biti korištene. Prvo će biti osvijetljene temeljne tehnologije, kao što su poslužiteljske aplikacije, virtualizacija te kontejnerizacija aplikacija. Nadalje, bit će razjašnjeni napredni okviri i biblioteke koji se koriste za implementaciju rješenja te opisana Cardano platforma na kojoj cijela ideja i počiva.

### **3.1. Poslužiteljska aplikacija**

Poslužiteljska aplikacija je program dizajniran za pružanje usluge ili resursa klijentskim aplikacijama ili uređajima. Radi na poslužitelju i odgovara na zahtjeve klijenata preko mreže. Poslužiteljske aplikacije obrađuju zadatke kao što su pohranjivanje podataka, obrada, komunikacija ili omogućavanje pristupa zajedničkim resursima. Slijede arhitekturu klijent-poslužitelj, gdje poslužitelj osluškuje dolazne klijentske zahtjeve i u skladu s tim ih obrađuje. Poslužiteljske aplikacije ključne su za omogućavanje komunikacije, suradnje i isporuke različitih usluga u umreženim okruženjima. U ovom rješenju, poslužiteljska aplikacija prima zahtjeve od drugih klijentskih aplikacija te ih na odgovarajući način obrađuje i proslijeđuje na Cardano čvor.

### 3.1.1. Izvršno okruženje

Node.js<sup>1</sup> je otvoreno, brzo i višeplatformsko izvršno okruženje za izvršavanje JavaScript koda. Node.js je JavaScript runtime izgrađen na V8 JavaScript engine-u, koji je razvio Google i koristi se u web preglednicima kao što je Google Chrome. Node.js omogućuje izvršavanje JavaScript koda izvan web preglednika, na poslužitelju ili na lokalnom računalu. To znači da razvojni inženjeri mogu koristiti JavaScript za pisanje poslužiteljske logike, rukovanje mrežnim zahtjevima, upravljanje datotekama, pristup bazama podataka i mnoge druge zadatke. Node.js je popularan izbor za razvoj pozadinskog sloja zbog svoje jedinstvene kombinacije prednosti. Prije svega, koristi JavaScript, široko prihvaćeni jezik kojim se pojednostavljuje razvojni proces. Također je važna i njegova sposobnost učinkovitog rukovanja asinkronim I/O operacijama bez blokiranja. To znači da Node.js može rukovati s višestrukim istodobnim vezama bez blokiranja izvođenja drugih operacija.

I/O model bez blokiranja, zajedno s V8 JavaScript engine-om, omogućuje visoko učinkovito izvršavanje koda. Ova arhitektura vođena događajima čini je posebno prikladnom za razvoj aplikacija u stvarnom vremenu, sustava za razgovor i drugih usluga gdje su odziv i skalabilnost ključni. Sve od navedenoga čini Node.js izvrsnim izborom za aplikacije koje zahtijevaju rukovanje velikim brojem istodobnih zahtjeva, kao što su API-ji i mikroservisi. Node.js može podnijeti veliku količinu istodobnih veza uz nisku potrošnju resursa. Štoviše, Node.js aplikacije mogu se lako vodoravno skalirati dodavanjem više poslužitelja za raspodjelu radnog opterećenja, omogućujući besprijekoran rast kako se potražnja korisnika povećava.

Node.js aplikacije mogu se implementirati na različitim platformama i uslugama u oblaku, što omogućuje fleksibilnost u odabiru infrastrukture. Podržava popularne platforme u oblaku kao što su AWS, Microsoft Azure i Google Cloud Platform-e, što olakšava posluživanje i skaliranje aplikacija u oblaku. Nadalje, Node.js ima koristi od opsežnog npm ekosustava. Node Package Manager (npm) je repozitorij paketa koji nudi golemu kolekciju biblioteka i modula otvorenog koda, što nevjerojatno olakšava integraciju postojećih rješenja u nove aplikacije. Node.js nudi izvrsnu podršku za rad s bazama podataka. Pruža razne biblioteke i module za interakciju s relacijskim (npr. MySQL, PostgreSQL) i NoSQL (npr. MongoDB, Redis) bazama podataka, omogućujući besprijekornu integraciju i učinkovito upravljanje podacima.

---

<sup>1</sup><https://nodejs.org/en>

### 3.1.2. Razvojni okvir

Express.js <sup>2</sup> široko je prihvaćen okvir za izradu poslužiteljskih aplikacija izgrađen na temelju Node.js-a, pružajući robustan skup alata i značajki za izgradnju učinkovitih i skalabilnih rješenja. Jedan od glavnih razloga za korištenje Express.js-a je njegova jednostavnost i minimalistički pristup. Nudi laganu strukturu koja omogućuje veću fleksibilnost i kontrolu nad arhitekturom aplikacija. Ideja razvoja je sljedeća: počni s malim i modularnim, postupno dodajući nove komponente i funkcionalnosti kako se projekt širi.

Express.js pruža jednostavan i intuitivan API koji olakšava definiranje ruta, rukovanje zahtjevima i upravljanje međuprogramima. Također ima bogat ekosustav modula, koji omogućuju dodavanje funkcionalnosti kao što su provjera autentičnosti, zapisivanje događaja i rukovanje pogreškama. Express.js promiče mogućnost ponovne upotrebe koda kroz svoj modularni dizajn, omogućujući organizaciju koda u zasebne module i komponente za višekratnu upotrebu. Takav modularni pristup olakšava održavanje i proširivanje aplikacije tijekom vremena.

S Express.js-om, moguće je izgraditi API-je, web usluge, pa čak i potpune web aplikacije. Pruža robusnu podršku za razvoj RESTful API-ja, što ga čini idealnim izborom za stvaranje pozadinskih usluga koje trebaju komunicirati s klijentima. Skalabilnost Express.js je još jedna prednost. Može učinkovito rukovati velikim brojem istodobnih veza, što ga čini prikladnim za aplikacije s velikim prometom ili zahtjevima u stvarnom vremenu. Express.js kompatibilan je i s različitim bazama podataka, SQL i NoSQL, omogućujući odabir baze podataka koja najbolje odgovara zahtjevima aplikacije.

Budući da Express.js pruža arhitekturu temeljenu na međuprogramima, omogućuje jednostavno dodavanje, uklanjanje ili modificiranje komponenti međuprograma kako bi se prilagodilo ponašanje aplikacije. Međuprogramski sustav je dobro osmišljen i intuitivan te omogućuje ulančavanje višestrukih međuprogramskih funkcija za obradu zahtjeva i odgovora. Takav sustav ulančavanja olakšava rad s HTTP protokolima i izvođenje uobičajenih operacija kao što je analiziranje tijela zahtjeva, rukovanje zaglavlјima i postavljanje kodova odgovora. Kako se mijenjaju zahtjevi aplikacija, moguće je jednostavno dodavati nove međuprograme bez potrebe za promjenom postojećih modula i programskog koda.

---

<sup>2</sup><https://expressjs.com/>

## 3.2. Cardano lanac blokova

Cardano lanac blokova je decentralizirana platforma s dokazom o udjelu treće generacije i dom je krypto valute Ada [10]. To je prva takva platforma koja se razvila iz znanstvene filozofije i pristupa koji je prvenstveno usmjeren na istraživanje. Platforma Cardano dizajnirana je od temelja i potvrđena od strane vodeće kombinacije vrhunskih inženjera i akademskih stručnjaka u područjima tehnologije lanaca blokova i kriptografije [16]. Snažan fokus je na održivosti, skalabilnosti i transparentnosti. Projekt je potpuno otvorenog koda koji ima za cilj isporučiti pravednu i otpornu infrastrukturu za financijske i društvene aplikacije na globalnoj razini. Jedan od primarnih ciljeva je pružiti pouzdane, sigurne financijske usluge onim ljudima koji trenutno nemaju pristup takvim uslugama, najviše u zemljama u razvoju.

Cardano je dizajniran sa sigurnošću kao jednim od temeljnih načela. Napisan je u funkcionalnom programskom jeziku Haskell u kojemu se potiče izgradnja sustava korištenjem čistih funkcija, što dovodi do dizajna u kojem se komponente mogu jednostavno testirati u izolaciji. Nadalje, napredne značajke programskog jezika Haskell omogućuju korištenje metoda za osiguravanje ispravnosti koda, kao što je temeljenje implementacije na formalnim i izvršnim specifikacijama, opsežno testiranje temeljeno na svojstvima i izvođenje testova u simulaciji [19]. Takav pristup je posebno važan u razvoju financijskih instrumenata. Cardano platforma za pametne ugovore (engl. *smart contract*) nastoji pružiti naprednije značajke od bilo kojeg prethodno razvijenog protokola i služiti kao stabilna i sigurna platforma za razvoj decentraliziranih aplikacija.

### 3.2.1. Mehanizam konsenzusa

Dokaz o udjelu vrsta je mehanizma konsenzusa ili protokola koji koristi količinu udjela koja se drži u sustavu za određivanje konsenzusa [18]. Konsenzusni protokol je ono što kontrolira zakone i parametre koji upravljaju ponašanjem lanca blokova. Konsenzus je time zapravo skup pravila kojih se pridržava svaki sudionik decentralizirane mreže. Budući da lance blokova ne kontrolira nijedno središnje tijelo, koristi se protokol konsenzusa kako bi se sudionicima distribuirane mreže omogućilo da se dogovore o povijesti mreže spremljenoj na lancu blokova, kako bi se postigao konsenzus o tome što se dogodilo i nastavilo iz jednog izvora istine.

Cardano je izgrađen na konsenzusnom protokolu Ouroboros i prvi je protokol koji je razvijen kroz recenzirana istraživanja [17]. U središtu protokola su skupovi uloga,

pouzđani čvorovi poslužitelji koje vodi operater tog uloga na koje drugi vlasnici kripto valute Ada mogu delegirati svoj ulog. Skupovi uloga koriste se kako bi se osiguralo da svi mogu sudjelovati u protokolu, bez obzira na tehničko iskustvo ili mogućnosti za održavanje čvora u radu. Ovi skupovi udjela usmjereni su na održavanje i držbe kombinirane udjele različitih dionika u jednom entitetu. Nasuprot tome, dokaz o radu je sinkroni protokol koji potiče rudare da se natječu tko će prvi riješiti probleme unutar bloka. Sustav nagrađivanja koristi se za poticanje ovog rješavanja problema. Međutim, ovaj pristup ima svoju cijenu, s povećanom potrošnjom električne energije i duljim vremenskim rasponima za rješavanje problema unutar lanca.

Jedna od ključnih značajki dokaza o udjelu je da kako se broj korisnika povećava, povećava se i mogućnost održavanja glavne knjige. To znači veću šansu za proizvodnju novih blokova koji se mogu dodati u lanac. Kreator novog bloka bira se na temelju kombinacije slučajnog odabira i određivanja njegovog uloga, odnosno bogatstva koje posjeduje. Vrsta izbora vođe događa se unutar lanca. Unutar protokola za dokaz udjela sudionici akumuliraju naknade za transakcije i na taj način povećavaju svoje bogatstvo kako vrijeme prolazi. Ovaj pristup potiče stalan i stabilan rast lanca blokova i smanjuje slučajeve zaustavljenih transakcija koje mogu spriječiti rast lanca i onemogućavanje provođenja transakcija.

Cardano ima cilj smanjiti centralizaciju te aktivno radi protiv ekonomskih poticaja koji bi vodili sustav prema centralizaciji. Čim postoje skupovi uloga, postoji i ekonomski poticaj da ti fondovi rastu, stoga je važno učiniti manje privlačnim da skup uloga postane prevelik [16]. Isplativije je imati mali broj velikih fondova nego veliki broj malih fondova. Cardano je dizajniran da radi protiv ekonomskih poticaja gdje veliki fondovi dominiraju sustavom, čineći manje privlačnim da neki fond postane prevelik. To je postignuto promjenom formule nagrađivanja. U naivnom sustavu, ukupne nagrade za skup bile bi proporcionalne njegovom ulogu, pa što je on veći, to bolje. S druge strane, ako skup privuče više uloga od određenog praga  $k$ , njegova nagrada više se neće povećavati. Dakle, ako svatko djeluje u vlastitom interesu kako bi maksimizirao svoje nagrade, onda se očekuje  $k$  skupova otprilike jednake veličine.

### **3.2.2. Model nepotrošenog izlaza transakcije**

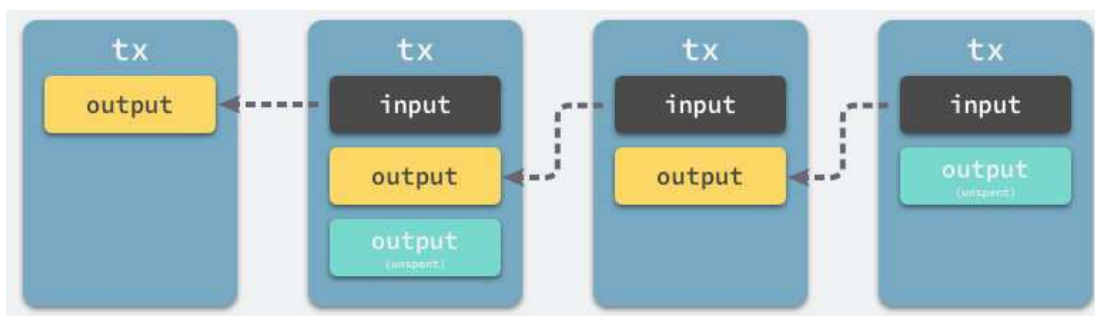
Cardano, poput Bitcoin lanca blokova, je temeljen na nepotrošenom izlazu transakcije (engl. *unspent transaction output*, *UTXO*). U takvom modelu, transakcija ima ulaze i izlaze, gdje su ulazi nepotrošeni izlazi iz prethodnih transakcija. Imovina se pohranjuje u glavnoj knjizi u nepotrošenim izlazima, a ne u računima [23, 11, 20]. Izlaz transak-



cije uključuje adresu i vrijednost, a potpis koji pripada adresi je ključ za otključavanje izlaza. Nakon otključavanja, izlaz se može koristiti kao ulaz. Nove transakcije troše izlaze prethodnih transakcija i proizvode nove izlaze koji se mogu potrošiti u budućim transakcijama.

Opisani proces ilustriran je na dijagramu 3.1. Kada je izlaz otključan ulazom, lanac blokova označava otključani izlaz kao potrošen. Cardanov EUTXO (engl. *extended unspent transaction output*, EUTXO) model nudi bolju skalabilnost i privatnost, kao i pojednostavljeniju transakcijsku logiku jer se svaki nepotrošeni izlaz može potrošiti samo jednom, što provjeru transakcije čini puno jednostavnijom [16]. Tako je moguć visok stupanj paralelizma jer čvor može paralelno potvrđivati transakcije ako te transakcije ne pokušavaju potrošiti isti ulaz.

Model nudi jedinstvene prednosti u odnosu na druge modele jer uspjeh ili neuspjeh validacije transakcije ovisi samo o samoj transakciji i njezinim ulazima. Kao posljedica toga, valjanost transakcije može se provjeriti izvan lanca, prije nego što se transakcija pošalje. Transakcija može ispasti nevažeća samo ako neka druga transakcija istodobno troši ulaz koji transakcija očekuje, ali ako su svi ulazi još uvijek prisutni, transakcija će zajamčeno uspjeti. To je u suprotnosti s modelom koji se temelji na računu kakav koristi Ethereum, gdje transakcija može propasti usred izvršavanja skripte [6]. Snažna značajka ovog modela također je da se naknade potrebne za valjanu transakciju mogu točno predvidjeti prije njezinoga objavljivanja. Ovo je jedinstvena značajka koja se ne nalazi u modelima koji se temelje na računu. Lanci blokova temeljeni na računu, poput Etheruma, nedeterministički su, što znači da ne mogu jamčiti objavu transakcije na lanac [23]. Ova neizvjesnost predstavlja rizik od novčanog gubitka i neočekivano visokih naknada.



**Slika 3.1:** Prikaz transakcija koje nastaju iz nepotrošenih izlaza prethodnih transakcija [1]

### 3.2.3. Izvorni tokeni

Izvorni tokeni (engl. *native token*) nova su značajka koja omogućuje transakcije više sredstava na Cardano lancu blokova. Korisnici mogu obavljati transakcije s kriptovalutom Ada i neograničenim brojem korisnički definiranih tokena. Izvorna podrška nudi različite prednosti za razvojne inženjere: nema potrebe za stvaranjem pametnih ugovora za rukovanje tokenima, što uklanja sloj dodatne složenosti i potencijal za pogreške. Lanac blokova koji podržava praćenje prijenosa i vlasništva nad različitim vrstama imovine zapravo ima podršku za više sredstava. U Cardano okruženju ovu funkcionalnost pruža značajka izvornih tokena. Za vrsta lanca blokova koji ima ugrađenu podršku za praćenje vlasništva i prijenosa više od jedne vrste imovine kaže se da ima izvornu podršku. Ako platforma podržava funkcionalnost pametnih ugovora, moguće je pratiti imovinu za koju ne postoji izvorna podrška, ali to se onda postiže pomoću rješenja drugog sloja i takva vrsta podrške onda nije izvorna.

Izvorni tokeni predstavljaju određenu vrijednost i djeluju kao obračunska jedinica koja se može koristiti za plaćanja, transakcije i može se poslati na novu adresu [7]. Iako i Ada i izvorni tokeni imaju vrijednost i djeluju kao jedinica za plaćanje i transakciju, samo se Ada koristi za plaćanje naknade i dobivanje nagrada [7]. Ovo svojstvo kriptovalute Ade posljedica je konstrukcije temeljnog protokola konsenzusa. Svaka adresa mora sadržavati minimalnu Ada vrijednost kako bi se mogli prenijeti drugi izvorni tokeni. Iz toga slijedi da je nemoguće stvoriti izlaze koji sadrže samo tokene.

Broj svake vrste tokena u izlazu ne utječe na minimalnu vrijednost izlaza, ali broj vrsta tokena sadržanih u izlazu povećava tu minimalnu vrijednost. Razlog za to je taj što nazivi i ID-evi svake vrste tokena zauzimaju dodatni prostor u izlazu. Na taj način, slanje tokena na adresu uvijek uključuje i slanje minimalne vrijednosti Ada kriptovalute na tu adresu. Prije prijenosa tokena, korisnici mogu koristiti komunikaciju izvan lanca kako bi pregovarali tko isporučuje potrebnu vrijednost Ada kriptovalute za pokrivanje troškova transakcije.

### 3.2.4. Politika kovanja tokena

Politika kovanja tokena (engl. *minting policy*) je skup pravila koja upravljaju kovanjem i spaljivanjem imovine na lancu blokova obuhvaćene tom politikom [7]. Smisao politike kovanja je određivanje uvjeta pod kojima se tokeni mogu kreirati (engl. *mint*) ili uništiti (engl. *burn*). Na primjer, pravila mogu odrediti tko ima kontrolu nad opskrbom imovinom putem kovanja i spaljivanja. Politike kovanja definiraju korisnici koji žele stvoriti novu imovinu. Na primjer, korisnik bi mogao poželjeti samo sebi dopustiti

da kreira određenu vrstu tokena. To bi bilo navedeno u politici kovanja. Pridržavanje pravila kovanja provjerava čvor u trenutku obrade transakcije, pokretanjem koda ili provjerom relevantnih potpisa. Transakcije se moraju pridržavati svih pravila kovanja svih sredstava koja se transakcijom pokušavaju kovati.

Sva imovina na lancu blokova nužno ima politiku kovanja. Na primjer, politika kovanja kripto valute Ada je da se nova Ada nikada ne može kovati [16]. Sredstvo nikada ne može promijeniti svoju pridruženu politiku kovanja. Drugim riječima, postojeći tokeni ne mogu se povezati s novom politikom. Korisnici, međutim, mogu otkupiti i spaliti sve postojeće tokene i kovati nove, s novom politikom kovanja. Ako je postojeće sredstvo u glavnoj knjizi obuhvaćeno određenom politikom, onda je zajamčeno da je izvorno iskovano u skladu s tom politikom. Veza između tokena i njihovih politika kovanja je trajna iz sigurnosnih razloga, takva značajka štiti korisnike i sustav od nelegitimno iskovanih tokena. Imovina povezana s različitim politikama kovanja nikad nije zamjenjiva jedna s drugom. Ako se politika kovanja tokena promijeni kroz vrijeme, to zapravo više nije isti token, jer jednostavno ne koristi jednaku politiku kovanja, i njegova se vrijednost ne može usporediti s onom izvornog tokena.

Politike kovanja mogu biti slijedeće:

1. Politika jednog izdavatelja
2. Vremenski zaključana politika kovanja
3. Politika jednokratnog kovanja

Politika kovanja jednog izdavatelja određuje da samo subjekt koji posjeduje određeni skup ključeva smije kovati tokene određene grupe sredstava. Na primjer, skup ključeva navedenih u politici kovanja mora potpisati transakciju kovanja. Primjer grupe sredstava koja bi koristila politiku jednog izdavatelja bili bi tokeni koji predstavljaju bejzbol kartice. Tvrtka koja proizvodi kolekcionarske kartice objavila bi ključeve potrebne za kovanje novih bejzbol kartica. To bi značilo da se nijedan novi token ne može kovati bez potpisa te tvrtke.

Vremenski zaključana politika kovanja može se koristiti za određivanje kada se tokeni mogu kovati. Na primjer, samo u određenom vremenskom intervalu ili nakon njega ili samo prije određenog vremena. Ova vrsta politike obično se ne koristi sama. Najčešće je u kombinaciji s politikom jednog izdavatelja. Na primjer, politika može glasiti da tvrtka iz prethodnog primjera može kreirati nove tokene samo kroz sljedeće dvije godine i nakon toga više neće biti moguće kreirati nove tokene.

U politici jednokratnog kovanja, kompletan skup tokena dane grupe imovine kovan je jednom specifičnom transakcijom. To znači da više nikada neće biti kovani tokeni u toj određenoj grupi sredstava. Ova vrsta politike zahtijeva dodatnu implementaciju preko pametnih ugovora. Politika jednokratnog kovanja bila bi korisna za generiranje ulaznica za koncerte. Kapacitet dvorane u kojoj će biti koncert poznat je unaprijed tako da neće biti potrebe dopustiti da se kreira više ulaznica naknadno.

Izdavatelji tokena izdaju nove tokene i održavaju rezervu tokena u optjecaju. Jednom kada se tokeni prestanu koristiti, mogu ih uništiti u skladu s politikom kovanja. Kada se iskuje novi token, ukupna ponuda tokena za taj token se povećava, ali nema utjecaja na ponudu kripto valute Ada. Ako je radi o nezamjenjivim tokenima, ukupna ponuda je jednostavno povećana na jedan komad. Izdavatelji tokena morat će zadržati nešto novca u svojim novčanicima kako bi platili naknade za te transakcije. Proces spaljivanja tokena jednostavno uklanja te tokene iz optjecaja, a ukupna ponuda tokena se smanjuje. Proces spaljivanja je identičan za zamjenjive i nezamjenjive tokene.

### 3.3. Kontejnerizacija programskog rješenja

Virtualizacija (engl. *virtualization*) i kontejnerizacija (engl. *containerization*) su dva različita pristupa implementaciji i upravljanju softverskim aplikacijama, svaki sa svojim vlastitim skupom karakteristika i slučajeva korištenja.

#### 3.3.1. Virtualizacija i kontejnerizacija

Virtualizacija, koja se koristi već nekoliko desetljeća, uključuje stvaranje više virtualnih strojeva (engl. *virtual machine*, *VM*) na jednom fizičkom poslužitelju. Svaki virtualni stroj pokreće vlastiti operativni sustav, što omogućuje izolaciju aplikacija i resursa. Virtualizacija pruža jaku izolaciju i sigurnost između virtualnih strojeva budući da su odvojeni od temeljnog hardvera. Međutim, virtualni strojevi dolaze s određenim troškovima u smislu korištenja resursa i vremena podizanja sustava zbog potrebe za oponašanjem hardvera.

Kontejnerizacija je, s druge strane, novija tehnologija koja se pojavila s usponom Dockera<sup>3</sup>. Docker spremnici enkapsuliraju aplikacije i njihove ovisnosti, omogućujući im dosljedan rad u različitim računalnim okruženjima. Za razliku od virtualnih strojeva, spremnici dijele jezgru operacijskog sustava, što ih čini laganima i bržima

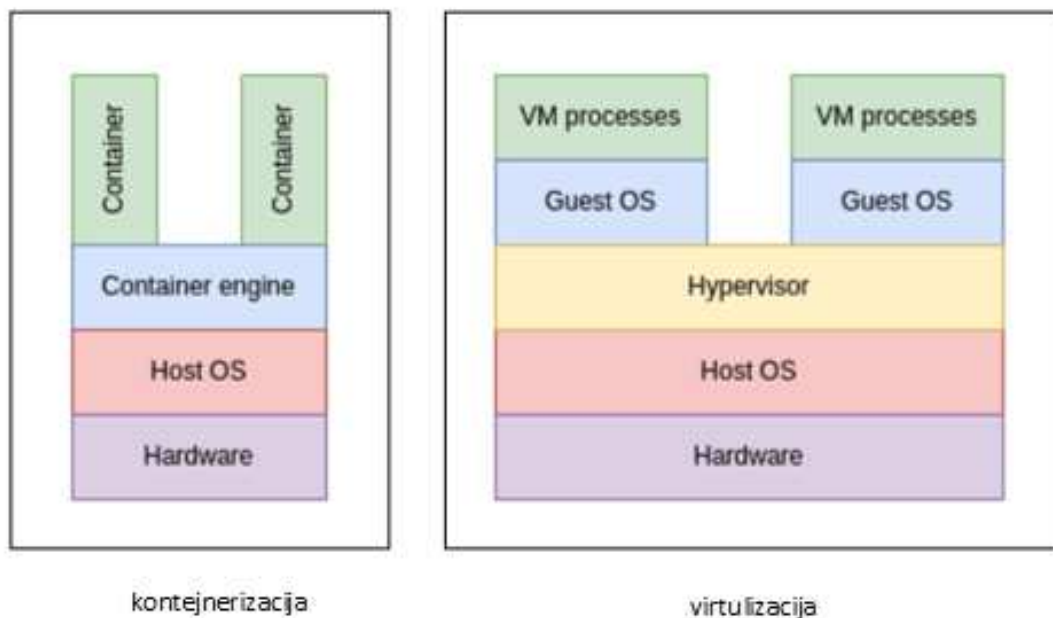
---

<sup>3</sup><https://www.docker.com/>

za pokretanje. Oni nude visoku prenosivost, skalabilnost i učinkovito korištenje resursa. Kontejnerizacija se temelji na konceptu slika koje sadrže sve što je potrebno za pokretanje aplikacije, uključujući kod, biblioteke i konfiguracije.

Prije pojave kontejnerizacije, virtualizacija je bila primarna metoda koja se koristila za pokretanje više aplikacija na jednom poslužitelju [14]. Omogućila je bolju iskoristivost hardverskih resursa pokretanjem više virtualnih strojeva, od kojih svaki pokreće vlastiti skup aplikacija. Virtualizacija je olakšala konsolidaciju radnih opterećenja, omogućujući organizacijama da smanje broj fizičkih poslužitelja i s njima povezane troškove. Omogućila je i učinkovit način upravljanja složenim infrastrukturama i osigurala izolaciju između aplikacija.

Međutim, virtualizacija je imala određene nedostatke. Zahtijevala je više resursa zbog pokretanja više operacijskih sustava i upravljanja njihovim zasebnim instancama [2]. Virtualni strojevi su se također sporije pokretali, što ih je činilo manje prikladnim za aplikacije koje je potrebno brzo skalirati. Dodatno, troškovi i složenost upravljanja virtualnim strojevima učinili su izazovnim postizanje brze implementacije i kontrole verzija. Kontejnerizacija je riješila ta ograničenja uvođenjem lagane, skalabilne i brže alternative. Kontejneri dijele jezgru OS-a domaćina, što rezultira smanjenim opterećenjem i bržim vremenom pokretanja. Opisana usporedba može se vidjeti na slici 3.2.



**Slika 3.2:** Razlika u arhitekturi kontejnerizacije (lijevo) i virtulizacije (desno)

Mogu se brzo osigurati i skalirati, što ih čini idealnim za moderne arhitekture mikroservisa i aplikacije u oblaku. Kontejnerizacija također pojednostavljuje implemen-

taciju aplikacije, budući da se kontejneri mogu lako pakirati, otpremati i raditi dosljedno u različitim okruženjima, od lokalnih razvojnih strojeva do proizvodnih poslužitelja.

U ovom programskom rješenju, kontejnerizacija je posebno zanimljiva zbog jednostavljenja prenosivosti razvijene aplikacije između raznih okruženja i poslužitelja. Također, omogućuje brzo skaliranje aplikacije, izolaciju grešaka i pojačanu sigurnost jer je aplikacija izolirana unutar spremnika te se tako sprječava da potencijalni zlonamjerni kod utječe na druge spremnike ili na sustav domaćina.

### **3.3.2. Docker i mikroservisi**

Docker je kontejnerska platforma otvorenog koda koja je revolucionirala način na koji se softverske aplikacije razvijaju, pakiraju i pokreću [15]. Uz Docker, aplikacije i njihove ovisnosti su enkapsulirane u lagane, prijenosne spremnike koji se mogu dosljedno izvoditi u različitim računalnim okruženjima. Spremnici se često nazivaju "lagani", što znači da dijele jezgru operacijskog sustava stroja. Spremnici pružaju samostalnu okolinu za izvođenje, uključujući kod aplikacije, biblioteke i konfiguracije, omogućujući besprijekorno i pouzdano izvođenje.

Docker pojednostavljuje proces implementacije aplikacije eliminirajući potrebu za brigom o razlikama u temeljnoj infrastrukturi [15]. Razvojni inženjeri mogu izgraditi Docker slike koje sadrže sve potrebne komponente i ovisnosti, osiguravajući dosljedno ponašanje bez obzira na glavni sustav. Te se slike mogu dijeliti, kontrolirati verzijom i lako reproducirati, smanjujući složenost implementacije. Ova prenosivost sprječava zaključavanje usluge u oblaku ili od strane nekog drugog ponuđača usluge te nudi druge značajne prednosti kao što su izolacija grešaka i izolacija sigurnosti. Kontejnerske aplikacije su "izolirane" po tome što se ne spajaju u kopiju operacijskog sustava. Umjesto toga, Docker se instalira na operativni sustav domaćina i postaje kanal za spremnike koji dijele operacijski sustavi s drugim spremnicima na istom računalnom sustavu.

Docker omogućuje brzo skaliranje i učinkovitost resursa dopuštajući da više spremnika radi na istom glavnom računalu, učinkovito iskorištavajući dostupne resurse [15]. Spremnici se mogu brzo pokretati i gasiti, omogućujući odgovore na promjenjive zahtjeve i neprimjetno skaliranje aplikacije. Docker također promiče arhitekturu mikroservisa, gdje se složene aplikacije rastavljaju na manje, labavo povezane usluge koje se mogu neovisno razvijati, testirati i implementirati u spremnike. Nove aplikacije temeljene na oblaku mogu se izgraditi iz temelja kao kontejnerske mikroservisi, razlažući

složenu aplikaciju na niz manjih specijaliziranih usluga kojima se može upravljati. Postojeće aplikacije mogu se prepakirati u spremnike koji učinkovitije koriste računalne resurse.

Mikroservisi se prihvaćaju kao superioran pristup razvoju i upravljanju aplikacijama, u usporedbi s ranijim monolitnim modelom koji kombinira softversku aplikaciju s pridruženim korisničkim sučeljem i bazom podataka u jedinstvenu jedinicu na platformi jednog poslužitelja [15]. S primjenom mikroservisa, složena aplikacija razbijena je u niz manjih, više specijaliziranih usluga, od kojih svaka ima svoju bazu podataka i svoju poslovnu logiku. Mikroservisi zatim međusobno komuniciraju preko uobičajenih sučelja (kao što su API-ji) i REST sučelja (kao što je HTTP). Koristeći mikroservise, razvojni timovi mogu se usredotočiti na ažuriranje određenih dijelova aplikacije bez utjecaja na nju kao cjelinu.

Koncepti koji stoje iza mikroservisa i kontejnerizacije slični su budućim da su obje prakse razvoja softvera koje u biti pretvaraju aplikacije u zbirke manjih usluga ili komponenti koje su prenosive, skalabilne, učinkovite i kojima je lakše upravljati. Štoviše, mikroservisi i kontejnerizacija dobro funkcioniraju kada se koriste zajedno. Spremnici pružaju laganu enkapsulaciju bilo koje aplikacije, bilo da se radi o tradicionalnom monolitu ili modularnom mikroservisu.

Današnje komunikacije brzo se sele u oblak gdje korisnici mogu brzo i učinkovito razvijati aplikacije [15]. Aplikacije i podaci temeljeni na oblaku dostupni su s bilo kojeg uređaja povezanoga s internetom, što članovima tima omogućuje daljinski rad i u pokretu. Pružatelji usluga u oblaku (engl. *cloud service provider*, *CSP*) upravljaju temeljnom infrastrukturom, što organizacijama štedi troškove poslužitelja i druge opreme, a također omogućuje automatizirano mrežno sigurnosno kopiranje za dodatnu pouzdanost [15]. Infrastrukture oblaka skaliraju se na zahtjev i mogu dinamički prilagoditi računalne resurse, kapacitet i infrastrukturu kako se zahtjevi opterećenja mijenjaju. Povrh toga, CSP-ovi redovito ažuriraju ponude, dajući korisnicima stalni pristup najnovijoj inovativnoj tehnologiji.

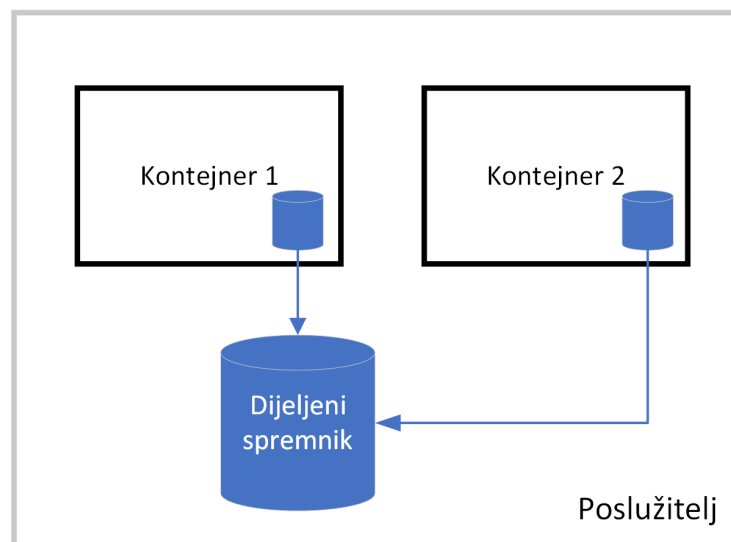
### **3.3.3. Komunikacija između procesa**

Komunikacija između procesa koji se izvode u različitim Docker spremnicima može se postići različitim metodama, ovisno o specifičnim zahtjevima i slučajevima korištenja. Neki od često korištenih pristupa su sljedeći:

- Mrežna komunikacija: Spremnici mogu međusobno komunicirati pomoću mrežnih protokola kao što su TCP/IP ili UDP. Svakom spremniku dodijeljena je IP

adresa unutar Docker mreže, što procesima unutar različitih spremnika omogućuje uspostavljanje veza i razmjenu podataka preko te iste mreže.

- Redovi poruka ili brokeri: Uvođenje reda poruka ili brokera, kao što je RabbitMQ, Apache Kafka ili Redis, može olakšati komunikaciju između spremnika. Spremnici mogu objavljivati poruke u redu poruka, a drugi spremnici mogu konzumirati te poruke. Ovaj asinkroni komunikacijski obrazac koristan je za odvajanje procesa i omogućavanje skalabilne komunikacije otporne na greške.
- Dijeljeni volumeni: Docker spremnici mogu dijeliti podatke koristeći dijeljene volumene. Zajednički volumen spojen je na više spremnika, omogućujući im čitanje i pisanje podataka na istu lokaciju na glavnom sustavu. Dijeljenjem volumena, procesi u različitim spremnicima mogu razmjenjivati datoteke ili pristupati zajedničkom izvoru podataka 3.3.



**Slika 3.3:** Dijeljenje zajedničkog prostora za pohranu između Docker kontejnera

U ovom rješenju komunikacija između Cardano čvora i razvijene aplikacije vrši se preko priključne utičnice, također poznata kao Unix domenska socket komunikacija. Takva vrsta komunikacije omogućuje procesima da uspostave komunikacijske kanale koristeći posebne socket datoteke umjesto mrežnih protokola. Kada programi žele komunicirati putem socket datoteka, obično slijede model klijent-poslužitelj. Poslužiteljski program stvara socket datoteku i veže je na određenu stazu u datotečnom sustavu. Klijentski program povezuje se s ovom datotekom utičnice pomoću putanje datoteke. Nakon što se veza uspostavi, programi mogu razmjenjivati podatke čitanjem



i pisanjem u deskriptor datoteke.

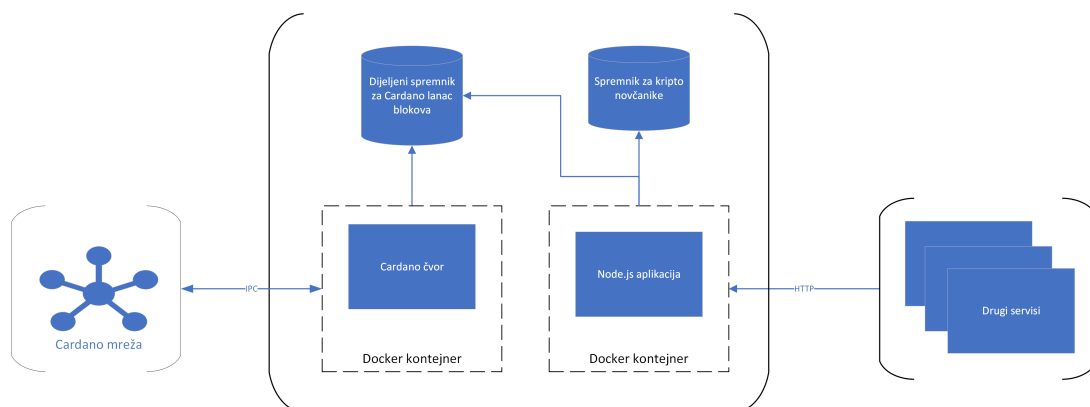
Programi prenose podatke na temelju unaprijed dogovorenoga komunikacijskog protokola, koji može biti jednostavan tekstualni protokol ili složeniji binarni protokol. Specifični format i značenje podataka određuju aplikacije uključene u komunikaciju. Komunikacija preko takve datoteke pruža lagan i učinkovit mehanizam za komunikaciju između procesa. Ostvarenje takve komunikacije realizirano je preko dijeljenog volumena između dva kontejnera. Dijeljeni volumen služi za pohranjivanje socket datoteke. Oba spremnika zatim mogu pristupiti socket datoteci preko tog spojenog volumena.

## 4. Implementacija programskog rješenja

Razvijeno programsko rješenje sastoji se od dvije glavne komponente: lokalnog Cardano čvora i poslužiteljske aplikacije koja stoji kao posrednik između Cardano lanca blokova i svojih klijenata. Poslužiteljska aplikacija omogućava svojim klijentima da preko HTTP protokola razmjenjuju podatke s Cardano lancem blokova. Ona translira jednostavne zahtjeve poslane preko HTTP protokola u naredbe razumljive Cardano protokolu. Uz to, u potpunosti apstrahira kreiranje i potpisivanje transakcija te općenito svu interakciju s krypto novčanicima. Važno je napomenuti da se za klijente podrazumijeva da se radi o drugim mrežnim aplikacijama, a ne fizičkim osobama. Razvijeno programsko rješenje je zamišljeno da se koristi unutar nekog sustava te da ima dobru mrežnu i fizičku zaštitu lokacije na kojoj se nalazi jer upravlja krypto valutom i tokenima od osobite važnosti.

### 4.1. Arhitektura programskog rješenja

Arhitektura cjelokupnog programskog rješenja prikazana je na slici 4.1. S lijeve strane nalazi se Cardano čvor, zapakiran u Docker kontejner. Podatci s čvora spremaju se u dijeljeni volumen, a čvor razmjenjuje poruke s javnom Cardano mrežom, odnosno lancem blokova, preko mrežnog protokola. Razvijena poslužiteljska aplikacija, također zapakirana u Docker kontejner, komunicira s lokalnim Cardano čvorom preko spomenutog dijeljenog volumena koristeći Unix priključnu utičnicu. Aplikacija svoje podatke, odnosno podatke krypto novčanika, sprema u poseban volumen samo za tu namjenu. Uz to, aplikacija prima HTTP zahtjeve od drugih servisa. Ti servisi se mogu nalaziti u drugim Docker kontejnerima ili na skroz odvojenim sustavima. Na taj način aplikacija služi kao posrednik za prenošenje zahtjeva svojih klijenata na i sa Cardano lanca blokova.



**Slika 4.1:** Prikaz arhitekture programskog rješenja

## 4.2. Pokretanje Cardano čvora i sinkronizacija

Kako bi se osigurala uspješna instalacija i bespriječno izvođenje aplikacije, ključno je uzeti u obzir hardverske specifikacije sustava. Preporučeni hardverski zahtjevi pokazuju da sustav treba imati minimalno 4 gigabajta (GiB) radne memorije (RAM) i najmanje 4 procesorske jezgre. Dodatno, kapacitet pohrane od 100 gigabajta (GiB) neophodan je za rad Cardano čvora. Međutim, važno je napomenuti da će se veličina Cardano lanca blokova povećati u budućnosti, stoga je preporučljivo dodijeliti više prostora za pohranu od navedenoga minimuma. Štoviše, veća količina radne memorije značajno poboljšava brzinu sinkronizacije lanca blokova, zbog čega je dobro imati i do 16 gigabajta (GiB) RAM-a. Ispunjavanjem ili premašivanjem ovih specifikacija hardvera, korisnici mogu osigurati optimalne performanse i izbjeći moguće probleme s kompatibilnosti tijekom korištenja aplikacije.

Za pokretanje programskog rješenja potrebno je najprije instalirati Docker alat. Docker pruža platformu za pakiranje i distribuciju programskog rješenja u izoliranom okruženju, te tako omogućuje izvođenje aplikacije bez potrebe za instalacijom dodatnih paketa i ručnim namještanjem postavki sustava. Samo aplikacija i Cardano čvor su zapakirani u zasebne Docker slike koje je potrebno pokrenuti te ih potom povezati preko dijeljenog spremnika kako bi programi međusobno mogli komunicirati i razmjenjivati poruke.

Za početak je potrebno napraviti tri direktorija. Na Unix sustavima to je moguće napraviti sa sljedećom naredbom:

```
mkdir {environment, cardano-data, wallet-data}
```

U kreirane direktorije spremat će se redom sljedeće vrste datoteka: postavke mreže

za Cardano čvor, Cardano lanac blokova - odnosno cijela povijest svih transakcija ikad nastalih na Cardano lancu blokova te podatci kripto novčanika.

Kako bi se Cardano čvor pravilno pokrenuo i kako bi se osiguralo da je spojen na pravu vrstu i verziju mreže, potrebno je preuzeti službenu konfiguraciju i pohraniti je u environment mapu. Pri pokretanju, Cardano će učitati konfiguraciju iz mape. U sljedećim primjerima, bit će korištena konfiguracija za testnet mrežu, točnije za pre-prod testnet vrstu mreže. Ta vrsta mreže je idealna za testiranje aplikacija na lancu blokova u uvjetima jednakima kao što su na samoj mainnet mreži, s jedinom razlikom u tome da nije potrebno trošiti pravu kripto valutu Ada pa samim time ni pravi novac. Korisnik može biti siguran da će se aplikacija koja je isprobana na pre-prod mreži ponašati jednako i u mainnet mreži.

Sve postavke za pre-prod mrežu mogu se preuzeti koristeći naredbu `curl`; kako slijedi:

```
curl -O -J https://book.world.dev.cardano.org/  
    ↪ environments/preprod/config.json  
curl -O -J https://book.world.dev.cardano.org/  
    ↪ environments/preprod/db-sync-config.json  
curl -O -J https://book.world.dev.cardano.org/  
    ↪ environments/preprod/submit-api-config.json  
curl -O -J https://book.world.dev.cardano.org/  
    ↪ environments/preprod/topology.json  
curl -O -J https://book.world.dev.cardano.org/  
    ↪ environments/preprod/byron-genesis.json  
curl -O -J https://book.world.dev.cardano.org/  
    ↪ environments/preprod/shelley-genesis.json  
curl -O -J https://book.world.dev.cardano.org/  
    ↪ environments/preprod/alonzo-genesis.json  
curl -O -J https://book.world.dev.cardano.org/  
    ↪ environments/preprod/conway-genesis.json
```

Proces za korištenje mainnet mreže je identičan, jedino je potrebno preuzeti druge konfiguracijske datoteke:

```
curl -O -J https://book.world.dev.cardano.org/  
    ↪ environments/mainnet/config.json  
curl -O -J https://book.world.dev.cardano.org/  
    ↪ environments/mainnet/db-sync-config.json
```

```

curl -O -J https://book.world.dev.cardano.org/
    ↪ environments/mainnet/submit-api-config.json
curl -O -J https://book.world.dev.cardano.org/
    ↪ environments/mainnet/topology.json
curl -O -J https://book.world.dev.cardano.org/
    ↪ environments/mainnet/byron-genesis.json
curl -O -J https://book.world.dev.cardano.org/
    ↪ environments/mainnet/shelley-genesis.json
curl -O -J https://book.world.dev.cardano.org/
    ↪ environments/mainnet/alonzo-genesis.json
curl -O -J https://book.world.dev.cardano.org/
    ↪ environments/mainnet/conway-genesis.json

```

Sada je moguće pokrenuti Cardano čvor u Docker kontejneru pomoću sljedeće naredbe:

```

docker run \
    --name cardano-node \
    --volume $PWD/environment:/configuration \
    --volume $PWD/cardano-data:/opt \
    inputoutput/cardano-node run \
        --config /configuration/config.json \
        --topology /configuration/topology.json

```

Naredba pokreće Docker kontejner imena cardano-node i pomoću parametara `--volume` spaja konfiguraciju iz mape `environment` koja se nalazi u trenutnom direktoriju s mapom `configuration` koja se nalazi unutar Docker kontejnera. Na taj način se Cardano čvoru omogućava da pročita preuzetu konfiguraciju za spajanje na mrežu. Drugi volumen na jednak način spaja podatke s Cardano mreže u lokalnu mapu `cardano-data` te tako pohranjuje cijelu povijest lanca blokova na lokalni disk. Ako se nekada kasnije zatvori Docker kontejner ili ponovno pokrene poslužitelj, svi podatci ostat će spremjeni na lokalnom disku. Kada se Docker kontejner ponovno pokrene, samo će učitati već postojeće datoteke i neće biti potrebno ponovno sinkronizirati cijeli lanac blokova.

Jednom kada je Cardano čvor pokrenut, početak će sinkronizaciju s ostatkom mreže. Taj proces može potrajati od nekoliko sati do nekoliko dana, ovisno o performansama sustava i brzini interneta. Nužno je da sinkronizacija završi u potpunosti prije nego li se aplikacija može početi koristiti. Status Docker kontejnera i same sinkronizacije moguće je pratiti preko sljedeće naredbe:

```
docker ps
```

koja daje ispis:

CONTAINER ID	IMAGE	NAMES
6625b5e6d143	inputoutput/cardano-node	cardano-node

i pogledom u zapise iz kontejnera:

```
docker logs -f cardano-node
```

koji daje sljedeći ispis:

```
CARDANO_BIND_ADDR=0.0.0.0
CARDANO_BLOCK_PRODUCER=false
CARDANO_CONFIG=/configuration/config.json
Listening on http://127.0.0.1:12798
CARDANO_DATABASE_PATH=/opt/cardano/data
CARDANO_LOG_DIR=/opt/cardano/logs
CARDANO_PORT=3001
CARDANO_SOCKET_PATH=/opt/cardano/ipc/socket
CARDANO_TOPOLOGY=/configuration/topology.json
Byron; Shelley
1.35.5
```

Iz ispisa je vidljivo koja je konfiguracija učitana, putanje na važne datoteke i utičnice, broj bloka koji se trenutno sinkronizira itd.

Nakon što je cijeli lanac blokova sinkroniziran s Cardano čvorom, u ispisu će to biti i vidljivo na sljedeći način:

```
Started opening Chain DB
Started opening Immutable DB
Validating chunk no. 567 out of 567. Progress: 99.82%
Validated chunk no. 567 out of 567. Progress: 100.00%
```

Nakon ovoga koraka, moguće je preći na sljedeći korak, instalaciju i pokretanje same aplikacije.

### 4.3. Instalacija aplikacije

Cijela aplikacija i svi potrebni paketi spakirani su u jednu Docker sliku koja se stvara iz priloženog Dockerfile-a. Dockerfile sadrži skup instrukcija koje je potrebno pokrenuti

kako bi se stvorila Docker slika u kojoj se nalazi aplikacija zajedno sa svim potrebnim komponentama. Dockerfile je prikazan u nastavku:

```
FROM node:lts-slim

ARG CARDANO_VERSION=1.35.5

ENV TZ=UTC
RUN ln -snf /usr/share/zoneinfo/$TZ /etc/localtime &&
    ↪ echo $TZ > /etc/timezone
RUN apt update && apt install --no-install-recommends -y
    ↪ wget

WORKDIR /opt/cardano-node
ENV CARDANO_VERSION=$CARDANO_VERSION
RUN wget --no-check-certificate \
    "https://update-cardano-mainnet.iohk.io/cardano-node-
    ↪ releases/cardano-node-$CARDANO_VERSION-linux.
    ↪ tar.gz"
RUN tar -xzf cardano-node-$CARDANO_VERSION-linux.tar.gz
RUN mv cardano-cli /usr/local/bin
WORKDIR /opt
RUN rm -rf cardano-node

WORKDIR /opt/cardano-middleware

COPY package.json .
COPY package-lock.json .

ENV NODE_ENV=production
ENV CARDANO_NODE_SOCKET_PATH=/opt/cardano/ipc/socket

RUN npm ci --omit=dev
COPY . .

VOLUME [ "priv" ]
```

## EXPOSE 3000

```
CMD ["npm", "start"]
```

Na početku skripte uzima se već spremna Docker slika s instaliranom Node.js aplikacijom. Upotreba ovakve slike slijedi sve dobre sigurnosne prakse kod izgradnje novih Docker slika. U nju se preuzima binarna datoteka `cardano-cli` verzije 1.35.5 koja je definirana na početku kao argument za izgradnju slike. Za korištenje neke druge verzije Cardano klijenta potrebno je promijeniti varijablu `CARDANO_VERSION`, a pri stvaranju slike bit će instalirana točna verzija. Nakon toga, pomoću `npm` naredbi instaliraju se svi potrebni paketi za produkcijsku verziju Node.js aplikacije, te se na kraju konačno pokreće poslužiteljska aplikacija.

Da bi se stvorila Docker slika naziva `cardano-middleware` iz navedenoga niza instrukcija, potrebno je pokrenuti sljedeću naredbu:

```
docker build --tag cardano-middleware:latest .
```

Nakon uspješno provedene akcije, moguće je vidjeti popis svih dostupnih Docker slika na sustavu pomoću sljedeće naredbe:

```
docker image ls
```

REPOSITORY	TAG	IMAGE ID	SIZE
<code>cardano-middleware</code>	<code>latest</code>	<code>6b7749da8e0d</code>	891MB
<code>inputoutput/cardano-node</code>	<code>latest</code>	<code>bba793bf23cd</code>	946MB

Jedino što je još preostalo je pokrenuti novo kreiranu Docker sliku i povezati je preko dijeljenog spremnika na drugi Docker kontejner u kojemu je pokrenut Cardano čvor. Preko tog dijeljenog spremnika vršit će se sva komunikacija između aplikacije i Cardano čvora. Drugi volumen koji se povezuje je `wallet-data` i služi za spremanje datoteka od kripto novčanika na lokalni disk. Dodatno se povezuju i vrata na kojima aplikacija sluša nadolazeće HTTP zahtjeve na vrata poslužitelja, tako da se aplikacijskom sučelju može pristupiti izravno preko vrata broj 80, što označava da se radi o standardnom HTTP protokolu. Moguće je, naravno, koristiti i neka druga dostupna vrata. Navedeno se radi sljedećom naredbom:

```
docker run \
  --name cardano-middleware \
  --publish 80:3000 \
```



```
--volume $PWD/cardano-data/cardano/ipc/socket:/opt/
    ↪ cardano/ipc/socket \
--volume $PWD/wallet-data:/opt/cardano-middleware/
    ↪ priv \
cardano-middleware:latest
```

Pri pokretanju Docker kontejnera vidljiv je sljedeći ispis iz aplikacije koji prikazuje verziju aplikacije i učitani profil, u ovom slučaju radi se o produkcijskom profilu. Također su vidljive i dodatne informacije, kao što su Cardano mreža na koju je aplikacija spojena te dostupni kripto novčanici. Postavke oko mreže i kripto novčanika bit će detaljno objašnjene u sljedećem poglavlju.

```
> cardano-middleware@1.0.0 start
> node src/index.js
```

```
{"level":"info","message":"Connecting through /opt/
    ↪ cardano/ipc/socket"}
{"level":"info","message":"Connected to testnet-magic 1"}
{"level":"info","message":"Loading wallets..."}
{"level":"info","message":"Creating wallet 'Wallet1'"}
{"level":"info","message":"Created wallet 'Wallet1'
    ↪ successfully"}
{"level":"info","message":"Creating wallet 'Wallet2'"}
{"level":"info","message":"Created wallet 'Wallet2'
    ↪ successfully"}
{"level":"info","message":"Creating wallet 'Wallet3'"}
{"level":"info","message":"Created wallet 'Wallet3'
    ↪ successfully"}
{"level":"info","message":"App running in production
    ↪ environment"}
```

Kako bi se provjerilo da aplikacija uistinu radi pravilno i ima mogućnost obrade HTTP zahtjeva, može se napraviti jednostavni HTTP GET zahtjev preko `curl` naredbe:

```
curl "http://127.0.0.1/api/v1/health"
```

Ako je sve ispravno pokrenuto i aplikacija je uspješno učitala sve postavke i ima mogućnost odgovora na klijentove zahtjeve, dobiva se sljedeći odgovor:

```
{"success":true,"status":"UP"}
```

Navedena HTTP ruta može se koristiti kao jednostavna provjera statusa aplikacije i može se implementirati na način da i vanjski sustavi automatizirano provjeravaju status. Budući da se naredba pokreće na istom poslužitelju na kojem je i Docker kontejner pokrenut, a aplikacija sluša na vratima 80, moguće joj je pristupiti preko adrese localhost na IP adresi 127.0.0.1. Na potpuno jednak način može se pristupiti aplikaciji i preko javne IP adrese poslužitelja tako da je potrebno dobro osigurati tko sve ima mrežni pristup ovoj i ostalim rutama.

## 4.4. Postavke aplikacije i kripto novčanika

Sve postavke vezane za aplikaciju i kripto novčanike mogu se pronaći u `src/config.js` konfiguracijskoj datoteci. U njoj se nalaze postavke od HTTP poslužitelja pa je tako moguće promijeniti zadana vrata na kojima aplikacija sluša zahtjeve, razinu opsežnosti ispisa, konfiguracijski profil te na kraju postavke vezane za Cardano čvor. Sve postavke moguće je definirati i u `.env` datoteci iz koje će one biti učitane pri pokretanju aplikacije. U slučaju da je datoteka prazna, bit će učitane zadane postavke koje su definirane u konfiguracijskoj datoteci. Pri pokretanju aplikacije u Docker kontejneru, varijabla `NODE_ENV` postavljena je na vrijednost `production`, vrata na vrijednost 3000, a `LOG_LEVEL` na razinu `debug` kako bi se i poruke najniže razine ispisivale na konzoli.

```
module.exports = {
  profile: process.env.NODE_ENV || 'development',
  port: process.env.PORT || 3000,
  logger: {
    logLevel: process.env.LOG_LEVEL || 'debug',
  },
  cardano: {
    socketPath: process.env.CARDANO_NODE_SOCKET_PATH,
    network: process.env.CARDANO_NETWORK || 'testnet-magic
    ↪ 1',
    wallets: ['Wallet1', 'Wallet2', 'Wallet3'],
  },
};
```

Postavke vezane uz sam Cardano čvor nalaze se u odjeljku pod nazivom `cardano`. Postavka `socketPath` definira lokaciju utičnice preko koje se vrši komunikacija s Cardano čvorom. Ta postavka je u Docker kontejneru već postavljena pri pokretanju na komunikaciju preko dijeljenog spremnika tako da ju nema potrebe mijenjati. Postavka odabira mreže postavljena je na zadanu vrijednost `testnet-magic 1` i odnosi se na `pre-prod` mrežu na koju je prethodno postavljen Cardano čvor preko preuzetih konfiguracijskih datoteka. Da bi se koristila neka druga vrsta mreže potrebno je promijeniti tu vrijednost, a sve moguće vrijednosti mogu se pronaći na službenoj Cardano stranici. U tom slučaju, potrebno je pokrenuti i u potpunosti sinkronizirati Cardano čvor na željenu mrežu. Za prelazak na `mainnet` potrebno je varijablu `CARDANO_NETWORK` postaviti na vrijednost `mainnet` u konfiguracijskoj ili `.env` datoteci.

Druga postavka vezana za Cardano čvor je odabir krypto novčanika. U polje `wallets` navode se imena novčanika koji se planiraju koristiti za plaćanje i potpisivanje transakcija. Minimalno jedan novčanik mora bit naveden i taj služi za potpisivanje transakcija, plaćanje naknada, a na njegovoj adresi nastaju kovani tokeni. Svi ostali novčanici mogu se koristiti isključivo za potpisivanje transakcija. Tako je na primjer moguće imati tri različite organizacije, od kojih svaka ima svoj krypto novčanik. Jedna organizacija se brine o tome da se na njenom novčaniku uvijek nalaze dostupna sredstva za plaćanje naknada, a ostale organizacije, odnosno njihovi novčanici, su tu kako bi transakcija bila važeća jer se smatra da je transakcija valjana jedino ako su se sve tri organizacije složile i potpisale tu transakciju.

Kroz vrijeme je, naravno, moguće tu listu organizacija i proširiti dodavanjem novih krypto novčanika. U danom primjeru koda postavljena su tri zadana novčanika imena `Wallet1`, `Wallet2` i `Wallet3`. Pri pokretanju aplikacije, učitava se konfiguracija novčanika, provjerava se postoje li već svi novčanici, te se stvaraju oni koji nedostaju. Pri inicijalnom pokretanju, kada nijedan novčanik još nije kreiran, tim procesom će oni biti stvoreni te će aplikacija započeti s radom. Primjer ispisa kod prvog pokretanja i stvaranja krypto novčanika vidljiv je ispod:

```
{"level":"info","message":"Loading  wallets ..."}
{"level":"info","message":"Creating  wallet  'Wallet1 '"}
{"level":"info","message":"Created  wallet  'Wallet1 '
  ↳ successfully"}
{"level":"info","message":"Creating  wallet  'Wallet2 '"}
{"level":"info","message":"Created  wallet  'Wallet2 '
  ↳ successfully"}
```

```

{"level":"info","message":"Creating wallet 'Wallet3'"}
{"level":"info","message":"Created wallet 'Wallet3'
  ↳ successfully"}
{"level":"info","message":"App running in production
  ↳ environment"}

```

## 4.5. Status i prijenos sredstava

Pri interakciji s bilo kojom vrstom lanca blokova, pa tako i lancem blokova Cardano, potrebno je znati određene informacije o stanju čvora. Jedna od informacija je status sinkronizacije, odnosno je li čvor sinkroniziran s ostatkom mreže ili nije, koji je zadnji blok koji je primljen itd. Te informacije se dalje mogu usporediti preko raznih platformi kako bi se zaključilo radi li sve ispravno.

Kako bi bili sigurni da je transakcija uspješno zapisana na lanac, potrebno je pogledati koliko je nastalo novih blokova otkad je transakcija zapisana u svoj blok. Ne postoji jednostavno pravilo za izračunati koliko blokova je potrebno za to, već se to radi statističkom procjenom vjerojatnosti. Ta vjerojatnost zapravo govori kolike su šanse da će trenutni lanac ostati važeći pa tako i transakcija koja je pohranjena u jedan od blokova u tom lancu. Što je više blokova u lancu, to su te šanse veće. Ukratko, svaki korisnik za sebe, ovisno o tome kako se odnosi prema riziku, mora procijeniti taj broj potrebnih blokova. Sve potrebne informacije za to mogu se dobiti preko sljedeće HTTP GET rute:

```
http://127.0.0.1/api/v1/node/info
```

Ova ruta prikazuje sve informacije dohvaćene s Cardano čvora, tako da je čvor moguće nadgledati direktno kroz aplikaciju. Isto kao što je slučaj i s prethodnom rutom, ova ruta može se implementirati u vanjskom sustavu za automatiziranu provjeru statusa Cardano čvora.

```

{
  "success": true ,
  "data": {
    "block": 1009780,
    "epoch": 73,
    "era": "Babbage",
    "hash": "db179d9dc5fb0e33c019af23408eef1aaebf"
  }
}

```

```

    "c7b18369d5d71e1fd3cbdf265707",
    "slot": 30112931,
    "syncProgress": "100.00"
  },
  "status": 200
}

```

Sve rute su implementirane na način da vraćaju HTTP statusni kod i logičku varijablu `success` preko koje se može doznati je li naredba uspješno izvršena u aplikaciji. Sva ostala informacijska polja nalaze se zapakirana u varijablu imena `data`.

Za prijenos sredstava na kripto novčanike i praćenje njihovog stanja, potrebno je znati adrese novčanika. HTTP GET ruta `wallet/address` vraća parove imena i pripadnih adresa svih novčanika. Obzirom da su u postavkama definirana tri novčanika, ruta vraća točno te novčanike naziva `Wallet1`, `Wallet2` i `Wallet3`. Stanje svakoga od njih vidljivo je preko rute `wallet/balance`. Početno stanje svakog novčanika je očekivano 0 Ada.

`http://127.0.0.1/api/v1/wallet/address`

```

{
  "success": true,
  "data": {
    "Wallet1": "
      ↪ addr_test1qzdf8eu7fas4u6y3jf77669sx2yrcreu7c2lkua
      6jjntalq08de9jgjqzpa9ql09jdrm4lpnxd425cq72nwgyn
      yfar6qtrjn0h",
    "Wallet2": "
      ↪ addr_test1qpcaffmmpsqqg9e7jhf7czz9l2nucuhey6rer387
      0xs0w7y30gul
      ke0jdghshyppue3dhfvhlpqfthmhxxv32atdenmrq9s5uk6",
    "Wallet3": "
      ↪ addr_test1qq6gtc77hz2ucjrd8mr dhcxh950zk2lhdkuhu8m
      va2zt55s654r
      yjuy5eqgmuy6djv2379k0cx4pee flmqt600hu230svmv9nt"
  },
  "status": 200
}

```

```
http://127.0.0.1/api/v1/wallet/balance
```

```
{
  "success": true ,
  "data": {
    "Wallet1 ": 0 ,
    "Wallet2 ": 0 ,
    "Wallet3 ": 0
  },
  "status ": 200
}
```

Novčanik je moguće napuniti prijenosom sredstava s bilo kojeg drugog novčanika koji sadrži kriptovalutu Ada ili direktnim prijenosom s neke od dostupnih kriptomjenjačnica. Obzirom da se ovdje radi o `pre-prod` mreži, a ne `mainnet` mreži, moguće je zatražiti testnu kriptovalutu Ada od Cardano organizacije. Testna kriptovaluta je u potpunosti besplatna i može se dobiti preko službene stranice <sup>1</sup>. Na slici je vidljiv primjer prijenosa sredstava na prvi kriptonovčanik s Cardano stranice. Odabrana mreža je Preprod Testnet i ta vrijednost odgovara postavkama u konfiguraciji.

Odmah nakon slanja zahtjeva dobiva se povratna informacija da je transakcija uspješna te da je testna kriptovaluta ADA poslana na adresu.

Nakon nekoliko minuta provjerom stanja novčanika kroz aplikaciju vidljivo je da se u prvom novčaniku sada nalazi točno 10000000000 Lovelace, što odgovara 10000 Ada (1 Ada = 1000000 Lovelace).

```
{
  "success": true ,
  "data": {
    "Wallet1 ": 10000000000 ,
    "Wallet2 ": 0 ,
    "Wallet3 ": 0
  },
  "status ": 200
}
```

---

<sup>1</sup><https://docs.cardano.org/cardano-testnet/tools/faucet/>

#### To request tokens using the faucet:

1. Enter the address of the account where you want to top up funds.
2. If you have been issued with an API key, please enter this to access any additional funds you may have been allocated.
3. Click **Receive test ada**.
4. Funds will be in the testnet account you specified within a few minutes.

#### Delegation

If you're an SPO and wish to test stake pool operations on preview or pre-production testnets, you can request some test ada to be delegated to your pool. For this, choose the environment you're working in, and select 'receive pool delegation' from the action menu.

Environment

Preprod Testnet

Action

Receive test ADA


Address \*


f77669sx2yrcreu7c2lkua6jjntalq08de9jgjqzpa9ql09jdrm4lpnxd425cq72nwgynyfar6qtrjn0h

The address to send funds to

API Key

Optional API key to bypass rate limiting

 I'm not a robot

  
reCAPTCHA  
Privacy - Terms

REQUEST FUNDS

**Slika 4.2:** Zatraživanje testne kripto valute Ada

#### Success

Your transaction has been successful and test funds have been sent to

**addr\_test1qzdf8eu7fas4u6y3jf77669sx2yrcreu7c2lkua6jjntalq08de9jgjqzpa9ql09jdrm4lpnxd425cq72nwgynyfar6qtrjn0h.**

Please verify the following transaction hash:

**68639388ef4b2730659b0098e3b8b6c74a0c8cb1e48909ba9e32dd8c3111a79d**

REQUEST MORE FUNDS

**Slika 4.3:** Rezultat zatraživanja testne kripto valute Ada

## 4.6. Kovanje nezamjenjivih tokena

Novi tokeni stvaraju se preko HTTP POST rute

`http://127.0.0.1/api/v1/token/mint`

U tijelo HTTP zahtjeva dodaje se JSON sa svim informacijama koje token mora sadržavati. Obavezna polja su `id` i `name`, u kojima su zapisani jedinstveni identifikator tokena i puno ime tokena. Ostala polja su opcionalna, ali ako postoje moraju biti u formatu kako slijedi:

```
{
  id: 'LandMine#42',
  name: 'MRUD',
  description: 'Directed fragmentation mine',
  latitude: '45.8001454',
  longitude: '15.970733',
  files: [
    {
      name: 'File #1024',
      mediaType: 'image/png',
      src: 'QmZP1qyDSdA32763LD4hAYW6xKp59XCAig4vo9hFuqtsL6',
    },
    {
      name: 'File #1025',
      mediaType: 'image/gif',
      src: 'LiZP1qyDSdA36763LD4hAYW6xGp59XCAig4vo9hOjqtsL7',
    },
  ],
}
```

- `id`: jedinstveni identifikator tokena, mora biti alfanumerički tip podatka, bez razmaka
- `name`: puni naziv, može sadržavati sve ASCII znakove
- `description`: opis, također može sadržavati sve ASCII znakove
- `latitude`: geografska širina na kojoj se nalazi naprava
- `longitude`: geografska dužina na kojoj se nalazi naprava



- `files`: polje datoteka koje su vezane uz napravu i njihovi izvori
  - `name`: ime datoteke
  - `mediaType`: MIME format datoteke
  - `src`: lokacija datoteke, može biti mrežna poveznica, preporuča se da pokazuje na neku vrstu decentralizirane pohrane

Kada poslužitelj zaprimi zahtjev za kovanjem novog tokena, prolazi kroz sljedeći skup koraka:

Kreiranje politike kovanja jednog izdavatelja koja određuje da samo subjekt koji posjeduje određeni skup ključeva smije kovati tokene ove prirode. To znači da samo kripto novčanici navedeni u konfiguraciji aplikacije mogu kreirati tokene ove vrste sada i u budućnosti.

```
{
  "scripts": [
    {
      "keyHash": "f67ccae63c95cc854058c0adf0e4743
e3d23f4cfb51b7e0d49aabbf8",
      "type": "sig"
    },
    {
      "keyHash": "6bf4c457421c553fdb3b8b2bb2257729
e2c08904ce8753d53c8e6fc0",
      "type": "sig"
    },
    {
      "keyHash": "e609d7795ef62e54ed4d15524a66be5f
de9768e2c204c82ad58376f2",
      "type": "sig"
    }
  ],
  "type": "all"
}
```

Kreiranje jedinstvenog identifikatora tokena iz heksadekadske vrijednosti varijable `id` i jedinstvene politike kovanja.

```
11ddf8ef671b9dba8b1d7ba65c696e8730594c617f1969f4bcf666f6
```

```

.
4c616e644d696e65233432

function createAssetId(assetName, policyId) {
  logger.info('Creating asset id from name and policy...')
    ↪ ;

  const assetNameHex = Buffer.from(assetName).toString('
    ↪ hex');
  const assetId = `${policyId}.${assetNameHex}`;

  logger.info(assetId);

  return assetId;
}

```

Obrada meta podatka iz tijela zahtjeva u format koji je moguće spremiti na lanac blokova, zajedno s kreiranim identifikatorom i politikom kovanja. Rezultat ove obrade je potpuno važeći JSON objekt koji definira strukturu izvornog nezamjenjivog tokena na Cardano lancu blokova. Takva vrsta tokena ima kodno ime 721 koje na Cardano lancu blokova govori da se radi o nezamjenjivom tokenu te omogućava Cardano protokolu da izvršava sve daljnje operacije s tim tokenom.

```

{
  "721": {
    "11ddf8ef671b9dba8b1d7ba65c696e87305
    94c617f1969f4bcf666f6": {
      "LandMine#42": {
        "name": "MRUD",
        "description": "Directed fragmentation mine",
        "files": [
          {
            "mediaType": "image/png",
            "name": "File #1024",
            "src": "QmZP1qyDSdA32763LD4hAYW6xKp
            59XCAig4vo9hFuqtsL6"
          },
          {

```

```

    "mediaType": "image/gif",
    "name": "File #1025",
    "src": "LiZP1qyDSdA36763LD4hAYW6xGp5
9XCAig4vo9hOjqtsL7"
  }
],
"latitude": "45.8001454",
"longitude": "15.970733"
}
}
}
}

```

Kreiranje transakcije radi se na način da se uzmu prethodno ne-potrošene transakcije iz primarnoga kripto novčanika (primarni kripto novčanik je prvi novčanik u popisu kripto novčanika u konfiguracijskoj datoteci) i upotrijebe se kao ulazne transakcije u novoj transakciji. Iz njih se dobije potrebna količina kripto valute Ada za plaćanje naknada kod slanja transakcije u mrežu. Izlazne transakcije iz nove transakcije čine ne-potrošeni ostatak ulaznih transakcija i jedan novi token količine jedan. Količina jedan odgovara tome da se radi upravo o nezamjenjivom tokenu. U varijabli `mint` koristi se prethodno kreirana skripta politike kovanja, a u varijabli `metadata` sve ostale informacije o novom tokenu. Varijabla `witnessCount` označava broj svjedoka transakcije, odnosno broj novčanika koji tu transakciju potpisuju.

Zadano je da svi dostupni novčanici potpisuju transakciju pa je tako taj broj postavljen na vrijednost duljine polja u kojemu su ti kripto novčanici navedeni. Iz tih informacija preko Cardano čvora kreira se nova transakcija. Na kraju je potrebno još samo ostaviti dovoljno kripto valute Ada za plaćanje naknada mreži. To se može napraviti na način da se dobije trenutno potrebna minimalna naknada na mreži s Cardano čvora te se onda ta količina oduzme od nepotrošenih ulaza u ovu transakciju. Cijeli proces rezultira time da izlazi iz nove transakcije sadrže jedan novi nezamjenjivi token i sve prethodno ne-potrošene transakcije izuzev male količine kripto valute Ada koja će biti potrošena na plaćanje naknade mreži. Naposljetku se iz tako formirane transakcije ponovno kreira transakcija preko Cardano čvora u svom izvornom formatu te se vraća kao rezultat opisane funkcije.

```

function buildTransaction(assetId, mintScript, metadata)
  ⇨ {

```

```

logger.info('Building transaction...');

const tx = {
  txIn: primaryWallet.balance().utxo,
  txOut: [
    {
      address: primaryWallet.paymentAddr,
      value: { ...primaryWallet.balance().value, [assetId]:
        ↪ 1 },
    },
  ],
  mint: [{ action: 'mint', quantity: 1, asset: assetId,
    ↪ script: mintScript }],
  metadata,
  witnessCount: wallets.length,
};

logger.info(tx);

const rawTx = cardano.transactionBuildRaw(tx);
const fee = cardano.transactionCalculateMinFee({
  ...tx,
  txBody: rawTx,
});

tx.txOut[0].value.lovelace -= fee;

logger.info(tx);

return cardano.transactionBuildRaw({ ...tx, fee });
}

Rezultat kreiranja transakcije:
{
  "metadata": {
    "721": {

```

```

"11ddf8ef671b9dba8b1d7ba65c696e8730594c617f1969
f4bcf666f6": {
  "LandMine#42": {
    "description": "Directed fragmentation mine",
    "files": [
      {
        "mediaType": "image/png",
        "name": "File #1024",
        "src": "QmZP1qyDSdA32763LD4hAYW6xKp59XCAig4
vo9hFuqtsL6"
      },
      {
        "mediaType": "image/gif",
        "name": "File #1025",
        "src": "LiZP1qyDSdA36763LD4hAYW6xGp59XCAig4
vo9hOjqtsL7"
      }
    ],
    "latitude": "45.8001454",
    "longitude": "15.970733",
    "name": "MRUD"
  }
}
},
"mint": [
  {
    "action": "mint",
    "asset": "11ddf8ef671b9dba8b1d7ba65c696e8730594c617f19
69f4bcf666f6.4c616e644d696e65233432",
    "quantity": 1,
    "script": {
      "scripts": [
        {
          "keyHash": "
↪ f67ccae63c95cc854058c0adf0e4743e3d23f4cf

```

```

        b51b7e0d49aabbf8",
        "type": "sig"
    },
    {
        "keyHash": "6bf4c457421c553fdb3b8b2bb2257729e2c0890
4ce8753d53c8e6fc0",
        "type": "sig"
    },
    {
        "keyHash": "e609d7795ef62e54ed4d15524a66be5fde9768e
2c204c82ad58376f2",
        "type": "sig"
    }
],
"type": "all"
}
}
],
"txIn": [
{
    "txHash": "964
↪ a5818d88f1c9379e37e4d881215e7b5b215ba310a
611a8231ec1716c33983",
    "txId": 0,
    "value": {
        "lovelace": 10000000000
    }
}
],
"txOut": [
{
    "address": "addr_test1qrm8ejhx8j2uep2qtrq2mu8ywslr6gl
5e763klstdfx4th7x9avkmduxfrdldqy4mcv4dl3xax
wpycja6md8whme6jdmqv6l4v",
    "value": {
        "11ddf8ef671b9dba8b1d7ba65c696e8730594c617

```

```

    f1969f4bcf666f6.4c616e644d696e65233432": 1,
    "lovelace": 9999790323
  }
},
"witnessCount": 3
}

```

Prije objave transakcije mreži potrebno je tu transakciju i kriptografski potpisati pa se to radi sa sljedećom funkcijom. Funkcija kao parametre uzima prethodno kreiranu transakciju u izvornom obliku i skriptu politike kovanja. Navigacijom kroz polje kripto novčanika, dohvaćaju se njihovi privatni ključevi koji su potrebni za kriptografski potpis. Preko naredbe Cardano čvoru transakcija se potpisuje. Za potpis transakcije potrebna je lista privatnih ključeva čiji broj sada odgovara postavljenom broju svjedoka iz prethodnog koraka, skripta politike kovanja i transakcija u izvornom obliku kreirana u prethodnom koraku.

```

function signTransaction(rawTx, mintScript) {
  logger.info("Signing transaction ...");

  const signingKeys = [];
  for (const wallet of wallets) {
    signingKeys.push(wallet.payment.skey);
  }

  const signedTx = cardano.transactionSign({
    signingKeys: signingKeys,
    scriptFile: mintScript,
    txBody: rawTx,
  });

  logger.info(signedTx);

  return signedTx;
}

```

Nakon svega navedenoga, transakcija se objavljuje mreži gdje je dalje Cardano validatori pohranjuju u novi blok. Klijentu se vraća sažetak poslana transakcije. Cijeli

proces je atomičan i sažetak nastaje jedino ako su svi prethodni koraci uspješno obavljani i transakcija je objavljena na mrežu. U slučaju da se to ne dogodi, transakcija jednostavno nije objavljena na mreži i novi token neće biti nikada stvoren. K tome, nikakve naknade nisu plaćene pa je stanje novčanika i cijelog lanca blokova jednako kao što je bilo i prije pokušaja stvaranja novog tokena. U tom slučaju, potrebno je ponoviti cijeli postupak na jednak način, ili preciznije, ponovno se koriste jednaki ulazi i meta podaci u transakciji.

```
{
  success: true,
  data: {
    transaction_hash: 'af4deef749aecc2f80311f21f472df74
7e8a5053b4685111d71c269f856f85ea',
    token_metadata: {
      name: 'MRUD',
      description: 'Directed fragmentation mine',
      latitude: '45.8001454',
      longitude: '15.970733',
      files: [
        {
          "mediaType": "image/png",
          "name": "File #1024",
          "src": "QmZP1qyDSdA32763LD4hAYW6xKp59X
CAig4vo9hFuqtsL6"
        },
        {
          "mediaType": "image/gif",
          "name": "File #1025",
          "src": "LiZP1qyDSdA36763LD4hAYW6xGp59X
CAig4vo9hOjqtsL7"
        }
      ],
    }
  },
  status: 200
}
```



## 4.7. Pregled rezultata

Kreirane tokene uvijek je moguće vidjeti u kripto novčaniku i pozivom već spomenutih funkcija, ali obzirom da su svi podatci na lancu blokova javno dostupni, to je moguće napraviti i na drugim Cardano čvorovima. Iz tog razloga moguće je sve informacije pregledati i na javno dostupnom pregledniku lanca blokova.

CardanoScan <sup>2</sup> je moćan i sveobuhvatan preglednik lanca blokova posebno dizajniran za Cardano. Korisnicima pruža jednostavno sučelje za istraživanje i analizu cijele Cardano mreže. CardanoScan nudi širok raspon značajki i funkcionalnosti koje korisnicima omogućuju pristup detaljnim informacijama o blokovima, transakcijama, adresama i tokenima na Cardano lancu blokova. Korisnici mogu jednostavno pretraživati i pratiti određene transakcije ili adrese, pregledavati statistike u stvarnom vremenu, pratiti mrežnu aktivnost i istraživati povijesne podatke. Dodatno, CardanoScan pruža potrebne uvide u ekosustav uloga, dopuštajući korisnicima da istraže skupove uloga, prate aktivnost delegiranja i prate nagrade za uloge.

The screenshot displays the CardanoScan (PreProd) web application. The top navigation bar includes links for Home, Blockchain, Metadata, Tokens, Pools, Certificates, and More. A search bar is present with the placeholder text 'Search transaction, address, block, epoch.slot, pool, stakeKey, policyId.assetName, f'. The main content area is titled 'Transaction Details' and shows the following information:

- Transaction Hash:** af4deef749aacc2f8031f21f472df747e8a5053b468511d71c269f856f85ee
- Block:** 1016729
- Assurance:** High (15744 confirmations)
- Epoch / Slot:** 73 / 381186
- Absolute Slot:** 38275586
- Timestamp:** 06/05/2023 11:53:06 AM
- Total Fees:** 0.209677
- Total Output:** 9,999.790323
- Certificates:** 0
- TTL:** 06/05/2023 2:39:02 PM (38285542)

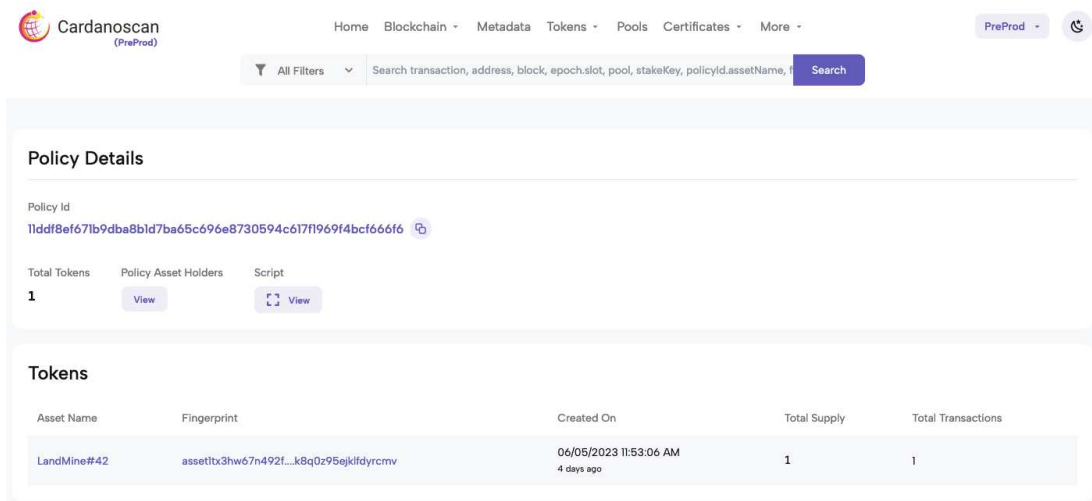
Below the transaction details, there is a tabbed interface with 'Summary', 'UTXOs', 'TokenMint (1)', and 'Metadata (1)'. The 'TokenMint (1)' tab is active, showing a table with the following data:

Policy Id	Asset Name	Fingerprint	Mint Amount
11ddf8ef671b9dba8b1d7ba65c696e8730594c617f1969f4bcf666f6	LandMine#42	asset1tx3hw67n492fh0cts7j5k8q0z95ejklfjdyrcmv	1

Slika 4.4: Prikaz detalja transakcije na pregledniku CardanoScan

Za pregled informacija o novom nezamjenjivom tokenu može se pretražiti transakcija `transaction_hash` iz prethodnog koraka na pregledniku CardanoScan. Na početnom ekranu 4.4 vidljive su osnovne informacije poput sažetka transakcije, bloka u kojemu se ona nalazi, koliko blokova je naknadno nastalo, vremenska oznaka itd.

<sup>2</sup><https://cardanoscan.io/>



**Slika 4.5:** Prikaz politike kovanja tokena na pregledniku CardanoScan

Obzirom da je nezamjenjivi token usko vezan uz politiku kovanja, do istih informacija moguće je doći i pretraživanjem politike kovanja 4.5. U tom slučaju izlistavaju se svi tokeni nastali baš tom politikom kovanja, a svi dodatni detalji sadržani u tokenu vidljivi su u polju s metapodacima 4.6.



**Slika 4.6:** Pregled metapodataka nezamjenjivog tokena na pregledniku CardanoScan

## 5. Zaključak

Zaključno, tehnologija lanca blokova riješila je problem dvostruke potrošnje te tako omogućila decentralizirane transakcije s digitalnim valutama poput Bitcoina. Njezina decentralizirana priroda i transparentni sustav raspodijeljene glavne knjige promijenili su brojne industrije eliminirajući posrednike i omogućavajući konsenzus među sudionicima. Potencijalne primjene lanca blokova proširuju se na sektore kao što su upravljanje opskrbnim lancem, decentralizirane financije, zdravstvo, državne službe i prava intelektualnog vlasništva.

Jedan konkretan primjer primjene lanca blokova je u humanitarnom razminiranju koje igra ključnu ulogu u naporima obnove nakon ratnih sukoba. Korištenjem tehnologije lanca blokova, moguće je riješiti izazove povezane s pouzdanošću i sigurnosti pri pohrani osjetljivih podataka iz različitih izvora. Decentralizirana mreža čvorova osigurava pouzdanost podataka i aktera te potiče suradnju među višestrukim dionicima, poboljšavajući učinkovitost i djelotvornost operacija razminiranja.

Kao rezultat razvijeno je programsko rješenje za stvaranje nezamjenjivih tokena na lancu blokova Cardano. Takva vrsta rješenja služi kao posredna aplikacija, omogućujući drugim aplikacijama u raznim sustavima izravnu interakciju s Cardano čvorom. Korištenjem ovog rješenja, razne se aplikacije mogu neprimjetno integrirati s lancem blokova, iskorištavajući njegove prednosti dok se u isto vrijeme od njih u potpunosti apstrahiraju sve operacije vezane uz obavljanje transakcija i kriptografskih potpisa. Programsko rješenje dolazi zapakirano u Docker kontejner kako bi se eliminirala potreba za brigom o razlikama u arhitekturi, povećala razina sigurnosti i pristupa te ubrzalo skaliranje i učinkovitost potrošnje računalnih resursa.

# LITERATURA

- [1] Utxo model. URL [https://blog.csdn.net/liudaoqiang\\_tj/article/details/81545002](https://blog.csdn.net/liudaoqiang_tj/article/details/81545002).
- [2] Charles Anderson. Docker [software engineering]. *Ieee Software*, 32(3), 2015.
- [3] Lennart Ante. Non-fungible token (nft) markets on the ethereum blockchain: Temporal development, cointegration and interrelations. *Economics of Innovation and New Technology*, 2022.
- [4] Lennart Ante. The non-fungible token (nft) market and its relationship with bitcoin and ethereum. *FinTech*, 1(3), 2022.
- [5] Andreas M Antonopoulos. *Mastering Bitcoin: unlocking digital cryptocurrencies*. " O'Reilly Media, Inc.", 2014.
- [6] Andreas M Antonopoulos i Gavin Wood. *Mastering ethereum: building smart contracts and dapps*. O'reilly Media, 2018.
- [7] Guido Barbian i Florian Mellentin. The cardano proof-of-stake protocol “ouroboros”.
- [8] Davi Pedro Bauer. Erc-721 nonfungible tokens. U *Getting Started with Ethereum: A Step-by-Step Guide to Becoming a Blockchain Developer*. Springer, 2022.
- [9] Amiangshu Bosu, Anindya Iqbal, Rifat Shahriyar, i Partha Chakraborty. Understanding the motivations, challenges and needs of blockchain software developers: a survey. *Empirical Software Engineering*, 24, 08 2019. doi: 10.1007/s10664-019-09708-7.
- [10] Lars Brünjes i Murdoch J Gabbay. Utxo-vs account-based smart contract blockchain programming paradigms. U *Leveraging Applications of Formal Methods*,

*Verification and Validation: Applications: 9th International Symposium on Leveraging Applications of Formal Methods, ISoLA 2020, Rhodes, Greece, October 20–30, 2020, Proceedings, Part III* 9. Springer, 2020.

- [11] V Buterin. Ethereum whitepaper: A next-generation smart contract and decentralized application platform [white paper], 2013.
- [12] Vitalik Buterin. What proof of stake is and why it matters. *Bitcoin Magazine*, 26, 2013.
- [13] Vitalik Buterin. Daos, dacs, das and more: An incomplete terminology guide. *Ethereum Blog*, 6, 2014.
- [14] Theo Combe, Antony Martin, i Roberto Di Pietro. To docker or not to docker: A security perspective. *IEEE Cloud Computing*, 3(5), 2016.
- [15] Inc Docker. Docker. *linea*].[Junio de 2017]. Disponible en: <https://www.docker.com/what-docker>, 2020.
- [16] Thomas Kerber, Aggelos Kiayias, Markulf Kohlweiss, i Vassilis Zikas. Ouroboros cryptosinous: Privacy-preserving proof-of-stake. U *2019 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2019.
- [17] Aggelos Kiayias, Alexander Russell, Bernardo David, i Roman Oliynykov. Ouroboros: A provably secure proof-of-stake blockchain protocol. U *Advances in Cryptology–CRYPTO 2017: 37th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 20–24, 2017, Proceedings, Part I*. Springer, 2017.
- [18] Sunny King i Scott Nadal. Ppcoin: Peer-to-peer crypto-currency with proof-of-stake. *self-published paper*, August, 19(1), 2012.
- [19] Pablo Lamela Seijas i Simon Thompson. Marlowe: Financial contracts on blockchain. U *Leveraging Applications of Formal Methods, Verification and Validation. Industrial Practice: 8th International Symposium, ISoLA 2018, Limassol, Cyprus, November 5-9, 2018, Proceedings, Part IV* 8. Springer, 2018.
- [20] Daniel Larimer. Transactions as proof-of-stake. *Nov-2013*, 909, 2013.
- [21] Fabian Schär. Decentralized finance: On blockchain-and smart contract-based financial markets. *FRB of St. Louis Review*, 2021.

- [22] Qin Wang, Rujia Li, Qi Wang, i Shiping Chen. Non-fungible token (nft): Overview, evaluation, opportunities and challenges. *arXiv preprint arXiv:2105.07447*, 2021.
- [23] Gavin Wood et al. Ethereum: A secure decentralised generalised transaction ledger. *Ethereum project yellow paper*, 151(2014), 2014.
- [24] Aviv Zohar. Bitcoin: under the hood. *Communications of the ACM*, 58(9), 2015.

## **Nezamjenjivi tokeni temeljeni na Cardano lancu blokova**

### **Sažetak**

Tehnologija lanca blokova transformirala je područje digitalnih valuta omogućivši decentralizirane transakcije i riješivši problem dvostruke potrošnje. Transparentna i decentralizirana priroda lanca blokova ima potencijal donijeti značajne promjene u industrijama kao što su upravljanje opskrbnim lancem, decentralizirane financije, zdravstvo, državne službe i prava intelektualnog vlasništva. U području humanitarnoga razminiranja osigurava pouzdanost i sigurnost podataka koji mogu dolaziti iz raznih izvora. Kako bi se olakšala integracija s Cardano čvorom, razvijeno je programsko rješenje koje omogućuje stvaranje nezamjenjivih tokena na Cardano lancu blokova. Takvo rješenje služi kao posrednik između drugih aplikacija i Cardano lanca blokova te pojednostavljuje složene transakcijske procese i kriptografske potpise. Kako bi se osigurala kompatibilnost, sigurnost, skalabilnost i učinkovitost korištenja resursa, aplikacija je zapakirana u Docker kontejner. Utjecaj lanca blokova obuhvaća više sektora, nudeći decentralizirana i transparentna rješenja koja potiču inovacije i poboljšavaju operativnu učinkovitost.

**Ključne riječi:** lanac blokova; Cardano; nezamjenjivi token; NFT; dokaz o udjelu

## **Non-fungible tokens based on Cardano blockchain**

### **Abstract**

Blockchain technology has transformed the field of digital currencies by enabling decentralized transactions and solving the problem of double spending. The transparent and decentralized nature of blockchain has the potential to bring about significant change in industries such as supply chain management, decentralized finance, healthcare, government services and intellectual property rights. In the field of humanitarian demining, it ensures the reliability and security of data that can come from various sources. In order to facilitate the integration with the Cardano node, a software solution was developed that enables the creation of non-fungible tokens on the Cardano blockchain. Such a solution serves as a middleware between other applications and the Cardano blockchain and simplifies complex transaction processes and cryptographic signatures. In order to ensure compatibility, security, scalability and efficient use of resources, the application is packaged in a Docker container. Blockchain's impact spans multiple sectors, offering decentralized and transparent solutions that drive innovation and improve operational efficiency.

**Keywords:** blockchain; Cardano; non-fungible token; NFT; proof-of-stake