Analiza decentraliziranih sustava za pohranu podataka

Jaković, Anđela

SVEUČILIŠTE U RIJECI
TEHNIČKI FAKULTET
Diplomski studij računarstva

Diplomski rad

Analiza decentraliziranih sustava za pohranu podataka

Rijeka, studeni 2024.

Anđela Jaković
0069085559

SVEUČILIŠTE U RIJECI
TEHNIČKI FAKULTET
Diplomski studij računarstva

Diplomski rad

Analiza decentraliziranih sustava za pohranu podataka

Mentor: prof. dr. sc. Kristijan Lenac

Rijeka, studeni 2024.

Anđela Jaković
0069085559

Umjesto ove stranice umetnuti zadatak za završni ili diplomski rad

Izjava o samostalnoj izradi rada

Ja, Anđela Jaković, izjavljujem da sam diplomski rad pod naslovom Analiza decentraliziranih sustava za pohranu podataka izradila samostalno pod mentorstvom prof. dr. sc. Kristijana Lenca.

Rijeka, studeni 2024.

_____

Anđela Jaković

Zahvala

Zahvaljujem mentoru prof. dr. sc. Kristijanu Lencu na podršci tijekom pisanja diplomskog rada, prenesenom znanju, uloženom trudu i vremenu.

Sadržaj

Popis slika

Popis tablica

Poglavlje 1

Uvod

U posljednjih nekoliko desetljeća napredak u tehnologiji rezultirao je značajnim promjenama u načinu na koji pohranjujemo, pristupamo i dijelimo podatke. Tradicionalni sustavi za pohranu podataka, koji se temelje na centraliziranim poslužiteljima, sve se češće suočavaju s izazovima u smislu sigurnosti, skalabilnosti i povjerenja. Naime, ti su sustavi podložni sigurnosnim provalama jer predstavljaju jednu točku kvara, što može dovesti do gubitka podataka ili ugrožavanja integriteta istih.

Za razliku od centraliziranih sustava, decentralizirani sustavi za pohranu podataka koriste distribuirane mreže kako bi omogućili pohranu podataka na više čvorova diljem mreže. U tom kontekstu blockchain tehnologija se pojavila kao inovativno konceptualno rješenje za decentraliziranu alternativu klasičnim metodama pohrane podataka. Stoga ovaj pristup pohranjivanju podataka dovodi do brojnih prednosti, uključujući povećanu otpornost na kvarove i napade te osigurava da su podaci uvijek dostupni, čak i kada neki čvorovi postanu nedostupni. Koristeći kriptografske metode za osiguranje integriteta i autentičnosti podataka, decentralizirani sustavi pružaju višu razinu sigurnosti, smanjujući rizik od neovlaštenog pristupa i manipulacije podacima.

Ovaj rad bavi se analizom decentraliziranih sustava za pohranu podataka temeljenih na blockchain tehnologijama. Cilj je istražiti kako ovi sustavi funkcioniraju, koje su njihove prednosti i nedostaci te kako se mogu implementirati za različite primjene, a posebna pozornost bit će posvećena proučavanju i testiranju postojećih

javno dostupnih rješenja te njihova međusobna usporedba.

U idućih nekoliko poglavlja, objašnjena je važnost decentraliziranih sustava za pohranu podataka, opisan je problem koji se istražuje u ovom radu, te su navedeni kriteriji za odabir javno dostupnih rješenja koja će se uspoređivati. Slijedi detaljan opis svakog odabranog rješenja, koja će se potom usporediti i analizirati u posljednjem poglavlju. Zaključak obuhvaća prednosti i nedostatke decentraliziranih sustava za pohranu podataka temeljenih na blockchain tehnologijama u odnosu na tradicionalne baze podataka.

Poglavlje 2

Uvod u decentralizirane sustave za pohranu podataka

Decentralizirani sustavi za pohranu podataka funkcioniraju na temelju distribuirane mreže čvorova, gdje svaki čvor predstavlja računalo spojeno na internet. Čvorovi pritom međusobno komuniciraju peer-to-peer, odnosno izravno, bez potrebe za autorizacijom na nekom centralnom poslužitelju. Kada korisnik pošalje podatke u decentralizirani sustav, ti podaci se dijele na manje dijelove koji se onda pohranjuju na različite čvorove unutar mreže. Također, svaki fragment podataka replicira se i pohranjuje na više čvorova, što osigurava da podaci ostaju dostpuni čak i slučaju kvara jednog ili više čvorova, budući da postoje kopije na drugim čvorovima.

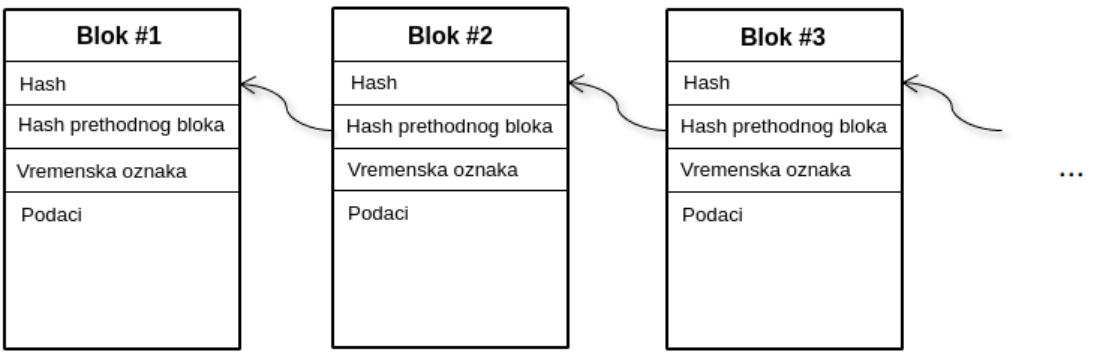
Prilikom spremanja podataka u bazu, korisnici svoje podatke enkrptiraju, čime se osigurava njihova povjerljivost i sigurnost. Samo korisnik s odgovarajućim kriptografskim ključem može dešifrirati i pristupiti originalnim podacima, čime se smanjuje rizik od neovlaštenog pristupa i manipulacije podacima. Podaci se grupiraju u fragmente i povezuju u blokove s kriptografskim hashom, jedinstvenim identifikatorom koji se koristi za provjeru integriteta podataka. Kada korisnik zatraži povrat podataka, fragmenti se prikupljaju i provjeravaju pomoću svojih hash vrijednosti kako bi se potvrdio identitet i osiguralo da originalan sadržaj nije izmijenjen. Proces je transparentan za korisnika i omogućava brzu i sigurnu povratnu informaciju.

Kako bismo bolje razumjeli arhitekturu i principe koji čine temelj takvih sus-

tava, potrebno je detaljnije se upoznati s blockchain tehnologijom, koja omogućuje decentraliziranost i integritet podataka, te IPFS protokolom, koji osigurava učinkovit prijenos i pohranu datoteka u distribuiranom okruženju. Ove tehnologije čine osnovu za mnoga moderna rješenja, koja će biti analizirana u kasnijim poglavljima ovog rada.

2.1 Blockchain

Blockchain je distribuirana knjiga (engl. *ledger*) u kojoj su podatkovni blokovi povezani u jednosmjerni lanac i u kojem svaki novi blok ovisi o vrijednosti prethodnog bloka. Svaki blok sadrži kriptografski hash prethodnog bloka, vremensku oznaku i podatke o transakciji (Slika 2.1). Početni blok u lancu još se naziva i "genesis block". Ako dođe do promjene podataka u nekom bloku, mijenja se i hash vrijednost tog bloka, što lančano utječe na sve naredne blokove u lancu. [1]



Slika 2.1 Struktura blockchaina

2.1.1 Svojstva blockchaina

Jedna od glavnih karakteristika blockchaina je to što je decentraliziran. Podaci su distribuirani na više čvorova (računala) diljem svijeta te svaki čvor njegove mreže

ima kopiju čitavog lanca blokova. Svi čvorovi mreže su jednako važni i komuniciraju direktno, odnosno peer-to-peer, bez centraliziranog posrednika.

Još jedno svojstvo blockchaina je njegova otvorenost i pristupačnost, što znači da je temeljen na otvorenom kodu i dostupan svima putem interneta, a za pristupanje nije potrebno jako računalo. Ovo svojstvo omogućuje korisnicima pristup s bilo kojeg mjesta u svijetu, pružajući mogućnost da proučavaju programski kod i sudjeluju u razvoju.

Neizmjenjivost blockchaina označava da se podaci unutar blockchaina ne mogu mijenjati niti brisati nakon što su jednom dodani u blokove. Kada se podaci unesu u blok, blok se povezuje s prethodnim blokom putem hash funkcije. Svaka promjena u podacima bi promijenila hash, što bi izazvalo neslaganje u cijelom lancu. Neizmjenjivost povećava povjerenje u točnost i pouzdanost podataka pohranjenih na blockchainu.

Još jedno svojstvo je programabilnost, odnosno sposobnost izvršavanja koda, često putem pametnih ugovora. Pametni ugovori omogućavaju automatizaciju transakcija i drugih procesa, što smanjuje potrebu za ljudskim resursima i troškove transakcija. Programabilni blockchaini omogućavaju stvaranje decentraliziranih aplikacija (dApps).

Posljednje svojstvo blockchaina koje će biti opisano u ovom poglavlju je samoodrživost. Samoodrživost se odnosi na sposobnost blockchaina da funkcionira i održava se bez potrebe za nadležnom osobom i administratorskim ovlastima. To znači da nitko ne može ugasiti sustav, niti isključiti nekog drugog korisnika. Mnoge blockchain mreže koriste mehanizme poput konsenzusnih protokola za upravljanje i održavanje mreže, a sudionici u mreži često su nagrađeni za svoj rad na održavanju blockchaina.

2.1.2 Konsenzus

Konsenzus protokol je postupak kroz koji svi sudionici blockchain mreže postižu zajednički dogovor o trenutnom stanju distribuiranog registra. U osnovi, konsenzus protokol osigurava da je svaki novi blok koji se doda u blockchain jedina verzija istine o kojoj se slažu svi čvorovi u blockchainu.

Različite blockchain mreže koriste različite metode konsenzusa kako bi se postigao dogovor o stanju pohranjenih podataka u distribuiranom sustavu. Postoji nekoliko mehanizama konsenzusa koji se koriste u različitim blockchain mrežama, a najpoznatiji su Proof of Work (PoW) i Proof of Stake (PoS).

Proof of Work ili dokaz o radu je prvi i najpoznatiji mehanizam konsenzusa, korišten u Bitcoinu i mnogim drugim kriptovalutama. U PoW sustavu, sva računala koja sudjeluju u mreži provjeravaju transakciju te se natječu u rješavanju složenog kriptografskog problema poput računanja hash funkcije. Prvo računalo koje riješi problem dobiva pravo dodati novi blok u blockchain i nagrađen je određenom količinom kriptovalute za uloženi rad. Ovaj proces zahtijeva značajnu računalnu snagu i energiju, a još se naziva i rudarenje.

Proof of Stake ili dokaz o udjelu je alternativni mehanizam konsenzusa koji koristi drugačiji pristup. U PoS sustavu sudjeluju validatori, korisnici koji su odgovorni za provjeru, validaciju i dodavanje novih blokova u blockchain. Validatorima se dodjeljuje pravo dodavanja novih blokova na temelju količine kriptovalute koju drže i "stavljaju na zalog". U ovom pristupu intezivni računalni rad i energija nisu potrebni, što čini PoS sustave energetski učinkovitijim. Validatori su potaknuti na pošteno ponašanje, jer mogu izgubiti svoj ulog ako pokušaju prevariti sustav.

Proof of Work zahtijeva značajnu računalnu snagu i energiju, što ga čini manje ekološki prihvatljivim. Unatoč tome, njegova otpornost na manipulacije i napade čini ga popularnim izborom za mnoge blockchain projekte. S druge strane, Proof of Stake koristi manje energije, ali se suočava s izazovima vezanim uz koncentraciju moći kod validatora koji posjeduju veću količinu kriptovalute, odnosno može doći do neželjene centralizacije.

Osim PoW-a i PoS-a, postoje i drugi mehanizmi konsenzusa poput Delegated Proof of Stake (DPoS), Practical Byzantine Fault Tolerance (PBFT) i Proof of Authority (PoA). Svaki od ovih mehanizama ima svoje prednosti i nedostatke te je prikladan za različite vrste blockchain mreža. [2]

2.1.3 Pametni ugovori

Pametni ugovori (engl. *smart contracts*) su samostalni programi pohranjeni na blockchainu koji automatski izvršavaju određene radnje kada su ispunjeni unaprijed definirani uvjeti. Programi omogućavaju automatizaciju složenih transakcija i poslovnih procesa, a time povećavaju sigurnost i transparentnost sustava.

Pametni ugovori funkcioniraju u nekoliko koraka:

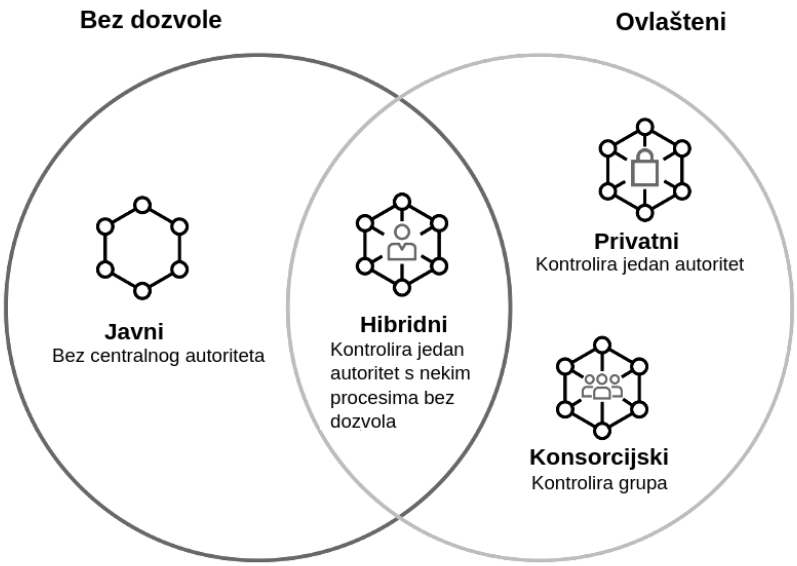
- **Dogovor**: Strane koje žele poslovati moraju se dogovoriti o uvjetima ugovora te kako će isti funkcionirati, kao i kriterije za njegovo izvršavanje.
- **Stvaranje ugovora**: Sudionici zatim kodiraju ugovor u programskom jeziku, bilo samostalno ili uz pomoć pružatelja pametnih ugovora. U ovoj fazi važno je osigurati sigurnost ugovora.
- **Implementacija**: Kada je ugovor izrađen, objavljuje se na blockchainu te tako postaje nepromjenjiv i nepovratan.
- **Praćenje uvjeta**: Pametni ugovor zatim prati blockchain, čekajući ispunjenje definiranih uvjeta. Ti okidači mogu biti bilo što što se može digitalno potrvditi (npr. uplata).
- **Izvršenje**: Kada su uvjeti ispunjeni, ugovor se automatski aktivira i izvršava definirane radnje (npr. prijenos sredstava).
- **Evidencija**: Rezultati izvršenja bilježe se na blockchainu, gdje su trajno pohranjeni i dostupni za pregled.

Prednosti pametnih ugovora uključuju veću učinkovitost, smanjene troškove, transparentnost i sigurnost. Budući da su pametni ugovori pohranjeni na blockchainu, ne mogu se mijenjati ili manipulirati nakon što su implementirani, što osigurava integritet i povjerenje u transakcije. [3]

2.1.4 Vrste blockchaina

Postoje različite vrste blockchaina, a svaka od njih ima specifične karakteristike koje ga čine pogodnim za različite primjene. Osnovna podjela blockchaina je na one

za koje "nije potrebna dozvola" (engl. *permissionless*) i "odobrene"/"ovlaštene" (engl. *permissioned*), ovisno o tome tko može izvršavati transakcije i tko ih može provjeriti. Blockchain za koji nije potrebna dozvola odnosi se na javne blockchain mreže, dok ovlašteni blockchain uključuje privatne i konsorcijske blockchain mreže. Osim osnovne podjele, postoji još i hibridni blockchain koji kombinira karakteristike javnih i privatnih blockchain mreža. Pregled svih vrsta blockchain mreža prikazan je na Slici 2.2.



Slika 2.2 Grafički prikaz vrsta blockchain mreža [4]

Javne blockchain mreže su potpuno decentralizirane i otvorene za svakoga tko želi sudjelovati. Svaki korisnik može postati čvor u mreži, sudjelovati u konsenzusu, validirati transakcije i pristupiti podacima pohranjenima na blockchainu. Ne postoji središnja točka kontrole. Sve odluke donosi zajednica korisnika i sve transakcije su javno dostupne i svatko ih može provjeriti. Javni blockchainovi se najčešće koriste u kriptovalutama, pametnim ugovorima i aplikacijama gdje je potrebna potpuna transparentnost i sigurnost.

Za razliku od javnih, privatni blockchainovi su kontrolirani od strane jedne organizacije ili grupe. Pristup mreži je ograničen, a sudionici moraju imati odobrenje

za sudjelovanje. Ove mreže su često centralizirane ili polucentralizirane, što znači da postoji jedna ili više centralnih točki kontrole. Budući da je broj sudionika manji i poznat, konsenzus se može postići brže nego u javnim mrežama. Privatni blockchainovi su idealni za korporativna okruženja, gdje tvrtke trebaju kontrolirati pristup podacima.

Hibridni blockchain omogućuje organizacijama da imaju kontrolu nad određenim podacima i transakcijama, dok istovremeno omogućuje da se određeni podaci javno dijele i provjeravaju. Hibridne blockchain mreže pružaju fleksibilnost, omogućujući organizacijama da zadrže privatnost tamo gdje je to potrebno, dok istovremeno koriste prednosti decentralizacije. U konsorcijskom blockchainu više organizacija upravlja mrežom umjesto jedne. Ove mreže su zatvorene za javnost, ali omogućuju više sudionika da sudjeluju u donošenju odluka i postizanju konsenzusa.

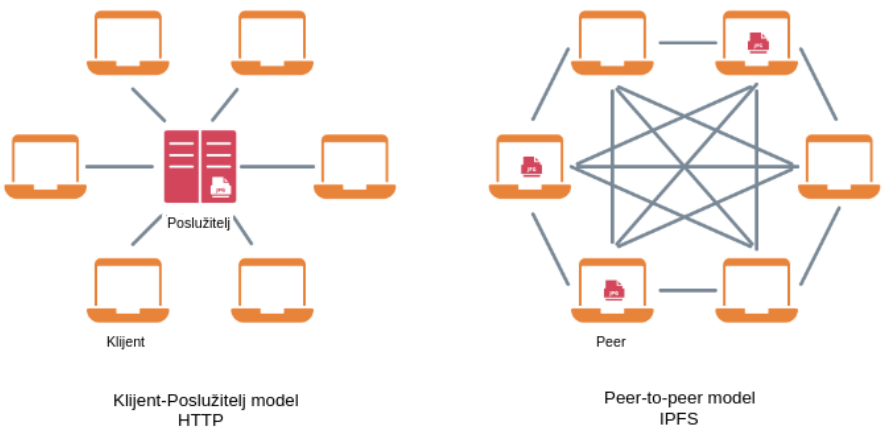
2.2 IPFS

InterPlanetary File System (IPFS) je decentralizirani sustav za pohranu i dijeljenje hipermedijskih sadržaja. IPFS koristi distribuiranu mrežu čvorova za pohranu podataka, gdje se podaci pohranjuju i dijele između čvorova u peer-to-peer mreži (Slika 2.3). Glavna ideja IPFS-a je decentralizacija, čime se omogućuje korisnicima da pohranjuju, preuzimaju i dijele datoteke bez potrebe za centraliziranim entitetom, poput servera ili pružatelja usluga u oblaku.

2.2.1 Struktura IPFS-a

IPFS koristi tehnologiju gdje svaki čvor u mreži može služiti kao domaćin podataka. Svaka datoteka u IPFS-u dobiva jedinstveni identifikator u obliku kriptografski generiranog hasha. Ovaj identifikator, poznat kao Content Identifier (CID), temelji se na sadržaju datoteke, što znači da svaka promjena u datoteci rezultira novim CID-om.

Datoteke pohranjene na IPFS-u podijeljene su u manje dijelove ili "blokove". Svaki od tih blokova ima svoj vlastiti CID, a svi blokovi su povezani kroz Merkle-DAG (Directed Acyclic Graph), što olakšava pronalaženje i dohvaćanje sadržaja s



Slika 2.3 Usporedba klijent-poslužitelj i peer-to-peer modela [5]

mreže. Kada čvor zatraži datoteku, on šalje zahtjev za blokove koji čine datoteku i prima ih od čvorova koji ih pohranjuju. Nakon što se dohvate svi blokovi, datoteka se može rekonstruirati. [6]

2.2.2 Prednosti i izazovi IPFS-a

Jedna od ključnih prednosti IPFS-a je njegova decentraliziranost, što smanjuje ovisnost o centraliziranim entitetima i povećava otpornost mreže na tehničke kvarove. Čak i ako određeni čvorovi postanu nedostupni, podaci ostaju dostupni na drugim čvorovima unutar mreže.

IPFS omogućuje veću efikasnost u pohrani i preuzimanju podataka zahvaljujući svojoj Merkle-DAG strukturi i CID-ovima. Korištenjem ovih tehnologija, IPFS smanjuje potrebu za pohranjivanjem istih podataka više puta, čime se smanjuje potrošnja resursa i poboljšava skalabilnost mreže. Osim toga, korisnici mogu preuzeti dijelove datoteka iz više izvora istovremeno, što značajno povećava brzinu preuzimanja.

Još jedna prednost IPFS-a je korištenje hash vrijednosti za generiranje CID-ova, što osigurava da korisnici uvijek preuzimaju točne i nepromijenjene podatke. Ova značajka također omogućuje verzioniranje datoteka. Svaki objekt (datoteka, direktorij ili podatak) u IPFS-u ima jedinstveni CID, koji je hash sadržaja. Kada

se sadržaj promijeni, njegov hash također se mijenja, što automatski stvara novu verziju sadržaja s novim CID-om. Pritom sadržaj nije nepovratno izgubljen, već sve verzije ostaju dostupne putem svojih CID-ova, što ovaj sustav čini jednostavnim za praćenje povijesti promjena i vraćanje na ranije verzije.

Međutim, način na koji sustav pohranjuje i replicira podatke može dovesti do redundancije podataka. Iako repliciranje podataka povećava dostupnost i otpornost, može uzrokovati veću potrošnju prostora za pohranu i smanjenje učinkovitosti.

Iako moćan, IPFS se susreće i sa određenim izazovima. IPFS može biti složen za implementaciju, posebno za korisnike koji nisu upoznati s distribuiranim sustavima i kriptografskim metodama. Osim toga, integracija IPFS-a s postojećim sustavima može zahtijevati dodatno prilagođavanje i optimizaciju. Također, brzina preuzimanja podataka u IPFS-u može varirati ovisno o dostupnosti čvorova i njihovoj fizičkoj udaljenosti. Iako IPFS omogućuje preuzimanje datoteka iz više izvora istovremeno, mreža može biti sporija u usporedbi s centraliziranim sustavima kada su čvorovi rijetki ili se nalaze daleko jedan od drugog.

Kombinacija distribuirane mreže, jedinstvenih hash vrijednosti i kriptografskih metoda čini IPFS moćnim alatom za decentraliziranu pohranu i dijeljenje podataka. Kao takav, IPFS ima širok spektar primjena, uključujući pohranu digitalnih sadržaja, dijeljenje datoteka, *hosting* web stranica i još mnogo toga.

Poglavlje 3

Metodologija

3.1 Ciljevi istraživanja

Cilj ovog istraživanja je analizirati i usporediti tri decentralizirana sustava za pohranu podataka: OrbitDB, IntegriDB i GroveDB. Glavni cilj je identificirati ključne značajke i karakteristike svakog sustava, njegove prednosti i nedostatke te koji sustav će najviše odgovarati potrebama i uvjetima. Usporedba se provodi na temelju niza definiranih kriterija, koji će biti opisani u nastavku ovog poglavalja.

3.2 Istraživački pristup

Za analizu i usporedbu odabranih decentraliziranih sustava primijenjen je kombinirani kvalitativni i kvantitativni istraživački pristup. Kvalitativni dio istraživanja obuhvatio je detaljnu analizu tehničke dokumentacije i primjera korištenja ovih sustava u stvarnom svijetu. Kvantitativni dio uključivao je prikupljanje i analizu podataka o performansama i troškovima kroz simulacije i dostupne metrike.

3.3 Kriteriji odabira sustava

Navedeni sustavi su izabrani radi njihove relevantnosti u području decentralizirane pohrane podataka, njihove popularnosti unutar blockchain zajednice te zbog specifičnih tehnoloških karakteristika koje omogućuju njihovu međusobnu usporedbu. Ovi sustavi predstavljaju različite pristupe u decentraliziranoj pohrani, od jednostavnih rješenja za pohranu podataka do složenih baza podataka sa širokim rasponom funkcionalnosti.

3.4 Kriteriji usporedbe

Usporedba je provedena prema sljedećim kriterijima:

- **Cijena pohrane**: Analizirani su troškovi pohrane podataka unutar svakog sustava, uključujući početne troškove, troškove održavanja te troškove koji nastaju zbog skalabilnosti i rasta volumena podataka.
- **Vrsta podataka**: Određene su vrste podataka koje svaki sustav pohranjuje, uključujući strukturirane i nestrukturirane formate te mogućnosti pohrane velikih datoteka ili metapodataka.
- **Način pohrane i pristupa**: Analizirane su mogućnosti pohrane različitih tipova podataka te kako svaki sustav podržava različite modele podataka.
- **Dostupnost dokumentacije i aktualnost održavanja**: Ispitana je kvaliteta i opseg dostupne dokumentacije, koliko se svaki sustav koristi u stvarnom svijetu, učestalost ažuriranja te koliko je aktivna zajednica koja razvija sustav.
- **Lakoća instalacije i korištenja**: Ispitano je je li kôd sustava otvoren (engl. *open-source*), evaluirana je jednostavnost procesa instalacije sustava i postavljanja mreže, kao i intuitivnost korisničkog sučelja.
- **Skalabilnost**: Evaluirana je sposobnost svakog sustava da učinkovito skalira s povećanjem volumena podataka i broja korisnika te kako se nosi s problemima poput latencije i opterećenja mreže.
- **Sigurnost i pouzdanost**: Analizirane su mjere sigurnosti i zaštite podataka,

uključujući enkripciju, kontrolu pristupa te zaštitu privatnosti korisnika.

- **Primjena**: Ispitano je za koje se vrste aplikacija koristi svaki sustav te koliko su prilagodljivi u različitim kontekstima (npr. pohrana za decentralizirane aplikacije, IoT, financijski sektor, itd.).

3.5 Postupak analize

Postupak analize obuhvaća evaluaciju ključnih karakteristika decentraliziranih sustava za pohranu podataka kroz definirane kriterije usporedbe. Detaljno su ispitane tehničke značajke svakog sustava, kao i praktični aspekti, uključujući kvantitativnu analizu skalabilnosti i performansi. Analiza je provedena kroz pregled dostupne tehničke dokumentacije, istraživanja povezanih radova i praktično testiranje sustava u kontroliranim uvjetima. Rezultati su korišteni za usporedbu sustava i identifikaciju njihovih prednosti i nedostataka u različitim kontekstima primjene.

Poglavlje 4

Pregled postojećih rješenja

Ovo poglavlje predstavlja pregled nekih od decentraliziranih rješenja za pohranu podataka koji su izabrani za kasniju usporedbu. Konkretno, radi se o tri sustava: OrbitDB, IntegriDB i GroveDB. Analizom ključnih značajki svakog od ovih rješenja, cilj je odrediti njihove prednosti i nedostatke te kontekst u kojem se svaki od njih može koristiti. Pregled postojećih rješenja poslužit će kao temelj za daljnu detaljnu analizu i usporedbu u kasnijim poglavljima ovog rada.

4.1 OrbitDB

OrbitDB je decentralizirana, peer-to-peer baza podataka izgrađena na protokolu IPFS, što znači da koristi mrežu distribuiranih čvorova za pohranu i razmjenu podataka. OrbitDB je dizajniran na način da omogući jednostavnu izradu aplikacija koje zahtijevaju sigurnu pohranu podataka u distribuiranom okruženju. Ova baza podataka se često koristi u kontekstu decentraliziranih aplikacija gdje je potrebno osigurati integritet i dostupnost podataka bez oslanjanja na centralizirane entitete. [7]

4.1.1 Arhitektura OrbitDB-a

OrbitDB koristi IPFS kao temeljnu mrežu za pohranu i razmjenu podataka. IPFS omogućuje distribuiranu pohranu podataka na više čvorova, čime se povećava otpornost na kvarove i napade. Svaki podatak pohranjen u OrbitDB-u ima jedinstveni identifikator (CID), koji omogućuje pristup podacima bez obzira na njegovu fizičku lokaciju.

Također, OrbitDB koristi log-based arhitekturu, što znači da se svi podaci pohranjuju u obliku zapisa (engl. *logs*) koji se kontinuirano nadograđuju. Svaki zapis je nepromjenjiv i može se povezati s prethodnim zapisima, čime se osigurava integritet i povijest podataka.

OrbitDB koristi CRDT (Conflict-free Replicated Data Types) strukture podataka koje omogućuju da se promjene u bazi podataka mogu distribuirati i replicirati bez rizika od konflikata. Više korisnika može simultano pisati i ažurirati podatke bez potrebe za posrednikom ili složenijim mehanizmima za rješavanje konflikata.

4.1.2 Vrste baza podataka u OrbitDB-u

OrbitDB podržava nekoliko različitih načina pohrane, od kojih svaka ima specifične karakteristike i primjene:

- **Key-Value Store**: Jednostavna baza podataka koja pohranjuje parove ključ-vrijednost. Idealna je za pohranu podataka gdje se svaki unos može identificirati jedinstvenim ključem.
- **Document Store**: Baza podataka dizajnirana za pohranu i pretraživanje dokumenata u JSON formatu. Korisna je za složenije strukture podataka i omogućuje filtriranje i pretraživanje po određenim poljima.
- **Event Log**: Baza podataka koja pohranjuje zapise događaja. Korisna je za aplikacije koje trebaju pratiti povijest događaja ili transakcija.
- **Counter Store**: Posebna vrsta baze podataka koja pohranjuje brojače te omogućuje sigurno i distribuirano zbrajanje vrijednosti.
- **Feed Store**: Baza podataka koja pohranjuje niz zapisa koji se mogu kronološki

pretraživati. Ova vrsta baze podataka često se koristi za pohranu vremenski osjetljivih podataka (npr. objave na društvenim mrežama).

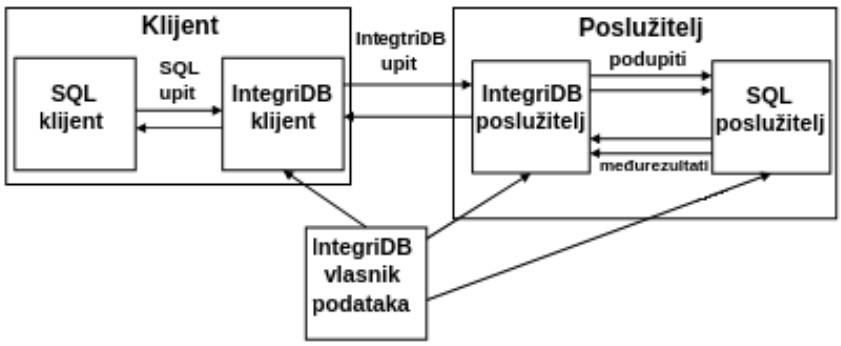
4.2 IntegriDB

IntegriDB je decentralizirana baza podataka osmišljena s ciljem da se pohrana podataka prepusti nepouzdanom poslužitelju, a zatim bilo tko može izvršavati provjerljive i složene SQL upite nad tom bazom podataka. Razvijen na temelju autentificiranih podatkovnih struktura (engl. *Authenticated Data Structures*, ADS), IntegriDB omogućava korisnicima da provjere točnost rezultata upita nad podacima pohranjenim na udaljenim serverima. Ovaj sustav je posebno koristan za aplikacije koje koriste SQL upite i zahtijevaju visoku razinu povjerenja u integritet podataka, a može se lako integrirati s postojećim SQL bazama podataka.

4.2.1 Arhitektura IntegriDB-a

IntegriDB koristi blockchain kao osnovu za pohranu podataka, pri čemu je svaki unos u bazu podataka kriptografski zaštićen i dodan u nepromjenjivi lanac blokova, čime se osigurava transparentnost i nepromjenjivost svih transakcija. Arhitektura sustava oslanja se na autentificirane podatkovne strukture koje omogućuju vlasnicima podataka pohranu na udaljenim serverima, dok klijentima pružaju mogućnost provjere točnosti tih podataka. ADS koristi Merkle stablo, gdje su listovi hashirani, a unutarnji čvorovi sadrže hash vrijednosti svojih potomaka, što omogućuje brzu i učinkovitu provjeru ispravnosti podataka.

IntegriDB se sastoji od nekoliko slojeva, uključujući klijentski sloj, sloj vlasnika podataka te serverski sloj. Klijent može biti i sam vlasnik podataka, koji koristi IntegriDB za slanje upita serveru. Server se sastoji od dva dijela: IntegriDB poslužitelja, koji se bavi kriptografskim operacijama, i SQL poslužitelja, koji izvršava SQL upite. IntegriDB poslužitelj generira dokaz o ispravnosti rezultata upita, koji klijent zatim provjerava prije nego što prihvati podatke. Struktura IntegriDB implementacije prikazana je na Slici 4.1.



Slika 4.1 Struktura IntegriDB implementacije [8]

4.2.2 Svojstva IntegriDB-a

Ključna prednost IntegriDB-a u odnosu na druga rješenja koja razmatramo u ovom radu što podržava širok spektar SQL funkcija, uključujući SUM, COUNT, AVG, MAX i MIN. Funkcija SUM omogućava provjeru zbroja vrijednosti u stupcu, dok se COUNT i AVG funkcije svode na SUM upite, pri čemu COUNT računa broj elemenata, a AVG prosjek vrijednosti. Funkcije MAX i MIN podržane su kao jednodimenzionalni rasponski upiti. Sustav također omogućava izvršavanje složenih upita poput JOIN upita nad rasponom, kao i ugniježđene upite koji kombiniraju ove funkcije. Međutim, određena ograničenja postoje u podršci za usporedbe i agregacije među stupcima, kao i u radu s duplikatima unutar JOIN upita.

Iako je usmjeren na decentraliziranu pohranu, IntegriDB nudi visoku razinu učinkovitosti, čak i pri radu s velikim količinama podataka. Sustav omogućava brzo provjeravanje rezultata (u milisekundama), dok serveru treba manje od minute za obradu, što je ključno za aplikacije koje zahtijevaju brzu obradu podataka.

Još jedna važna karakteristika IntegriDB-a je njegova skalabilnost. IntegriDB je skalabilan i može učinkovito upravljati velikim bazama podataka, uključujući tablice s milijunima redaka. Također, IntegriDB koristi enkripciju i hash funkcije, čime osigurava visoku razinu sigurnosti i povjerenja u sustav. [8]

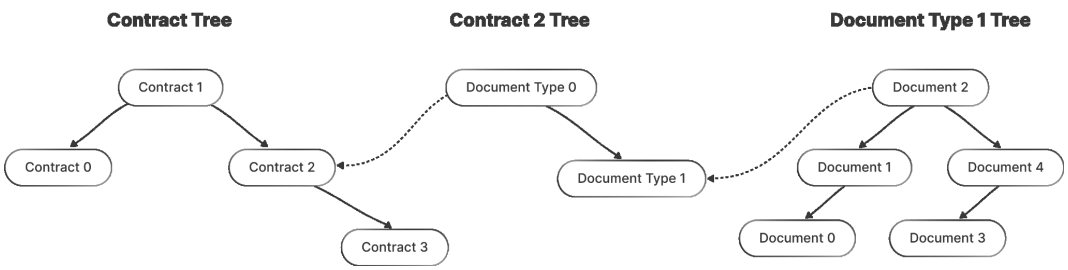
4.3 GroveDB

GroveDB je "šuma" Merkleovih stabala, gdje se korijenski hashovi stabala niže razine pohranjuju u čvorove stabala više razine, a korijenski hash korijenskog stabla može se koristiti za dokazivanje stanja cijele baze podataka. GroveDB je napredni sustav baze podataka osmišljen s naglaskom na učinkovito izvođenje upita putem sekundarnih indeksa, kriptografske dokaze, brzinu i pouzdanost. Napravljen je za korištenje unutar Dash platforme od strane Dash developera, ali se može lako integrirati u druge aplikacije za sličnu upotrebu.

4.3.1 Arhitektura GroveDB-a

HADS

GroveDB je prvi sustav koji implementira hijerarhijsku autentificiranu podatkovnu strukturu (engl. *hierarchical authenticated data structure*, HADS). HADS se može opisati kao skup autentificiranih podatkovnih struktura, poput Merkle stabala, koja sadrže dodatne ADS-ove. Primjer HADS arhitekture prikazan je na Slici 4.2. Ova arhitektura omogućuje učinkovito dokazivanje stanja baze podataka čak i za složene upite. Primjerice, u slučaju sekundarnih indeksa, HADS može stvoriti stabla za svaki sekundarni indeks, gdje svako stablo sadrži podstabla za jedinstvene vrijednosti. Stavke u tim podstablima mogu se odnositi na zapise u primarnom indeksnom stablu, čime se osigurava učinkovitost i smanjenje potrošnje memorije.



Slika 4.2 Contract tree [9]

Merk

Sva stabla unutar GroveDB-a su Merk stabla po defaultu. Merk stablo je specifična vrsta Merkle AVL stabla koja je optimizirana za performanse. Njegove ključne značajke uključuju brze operacije čitanja i pisanja, generiranje dokaza za opsege ključeva, podršku za paralelne operacije, replikaciju i kreiranje "checkpointova". Merk stabla omogućuju GroveDB-u postizanje visokih performansi čak i u okruženjima s ograničenim resursima, poput uređaja s niskom količinom RAM-a, dok istovremeno pruža skalabilnost potrebnu za rad na moćnim serverima.

4.3.2 Ključne značajke GroveDB-a

Sekundarni indeksi

Sekundarni indeksi omogućuju složenije i intuitivnije upite od onih temeljenih na primarnim indeksima. U sustavima bez sekundarnih indeksa, neki upiti zahtijevaju iteraciju kroz svaki zapis u bazi podataka, što značajno usporava rad. U GroveDB-u, sekundarni indeksi se koriste u kombinaciji s HADS-om kako bi omogućili učinkovitije pretrage i minimalnu potrošnju memorije, jer indeksi sadrže samo reference na stavke, a ne same stavke.

ACID svojstva

GroveDB podržava ACID (engl. *atomicity, consistency, isolation, durability*) svojstva koja se odnose na svojstva koja moraju biti ostvarena pri izvođenju transakcije, kako bi se osigurala valjanost podataka, čak i pri padu sustava. [10]

Radi se o sljedećim svojstvima:

- **Atomarnost**: U slučaju kada neka transakcija završi pogreškom, cijela transakcija prestaje, i baza ostaje nepromijenjena.
- **Konzistentnost**: Transkacije mogu dovesti bazu isključivo iz jednog validnog stanja u drugo validno stanje, odnosno izvođenje transakcije nikad neće srušiti bazu. To ne osigurava da su uneseni podaci točni, samo da su ispravno uneseni

u bazu.

- **Izolacija**: Svaka radnja je izolirana od druge, odnosno radnje s bazom koje su izvršavaju istovremeno daju isti rezultat kao da su radnje obavljene jedna iza druge.
- **Trajnost**: Sve završene transakcije s bazom ostat će zapisane i nepromijenjene u slučaju kvara.

Dokazi upita

Jedna od najznačajnijih inovacija koju donosi GroveDB je podrška za dokaze upita (engl. *query proofs*). Dokazi upita omogućuju korisnicima da verificiraju točnost rezultata dobivenih iz baze podataka, što je posebno važno u kontekstu decentraliziranih sustava.

Dokazi o uključenosti i odsutnosti

GroveDB koristi Merkle dokaze kako bi omogućio verifikaciju prisutnosti ili odsutnosti stavki u bazi podataka. Ovi dokazi osiguravaju da korisnici mogu pouzdano provjeriti stanje svojih podataka, bez potrebe za povjerenjem u operatere mreže.

Pohrana promjenjivih i nepromjenjivih podataka

GroveDB podržava pohranu i promjenjivih (engl. *mutable*) i nepromjenjivih (engl. *immutable*) podataka unutar iste baze. Ova fleksibilnost omogućuje da baza podataka odgovara širokom spektru upotrebe, od onih gdje su potrebni nepromjenjivi zapisi (npr. evidencija transakcija) do onih gdje su potrebne česte promjene i ažuriranja podataka.

Poglavlje 5

Usporedba i analiza

Postojeća rješenja decentraliziranih sustava za pohranu podataka koja su opisana u prethodnom poglavlju (OrbitDB, IntegriDB i GroveDB), u ovom poglavlju će biti analizirana i međusobno uspoređena prema unaprijed definiranim kriterijima: cijena pohrane, vrsta podataka, način pohrane i pristupa, dostupnost dokumentacije i aktualnost održavanja, lakoća instalacije i korištenja, skalabilnost, sigurnost i pouzdanost te primjena. Svaka od ovih baza podataka razlikuje se po nekim od navedenih kriterija te se prema tome koriste za različite primjene. U nastavku poglavlja je svaki kriterij detaljnije analiziran.

5.1 Cijena pohrane

Cijena pohrane je ključni kriterij kod uspoređivanja baza podataka, posebno u distribuiranim sustavima gdje decentralizacija, sigurnost i skalabilnost mogu dodatno povećati troškove. U tradicionalnim sustavima baza podataka, trošak pohrane uglavnom se sastoji od cijene hardverske infrastrukture, troškova održavanja i nadogradnje te raznih softverskih licenci. No u distribuiranim bazama podataka, kao što su OrbitDB, IntegriDB i GroveDB, dodatni faktori poput načina repliciranja podataka, transakcija i održavanja čvorova igraju važnu ulogu u ukupnom trošku pohrane. Distribuirane baze podataka koje rade na IPFS mrežama ili blockchain infrastrukturi općenito ovise o troškovima usluga trećih strana, mrežnim troškovima i energetskoj

efikasnosti mreže. Cijena pohrane podataka u takvim sustavima često nije fiksna i može se značajno razlikovati ovisno o veličini podataka, potrebnoj redundanciji i učestalosti pristupa podacima. Na taj način, cijena pohrane nije samo financijski trošak već i element koji utječe na korisničko iskustvo, pouzdanost sustava i održivost arhitekture u određenim uvjetima.

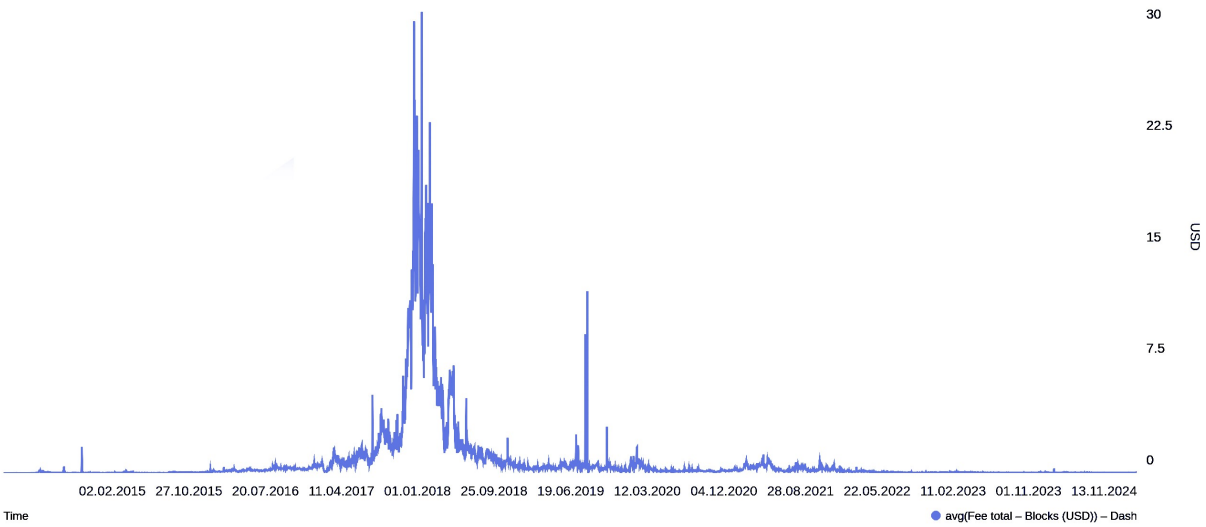
OrbitDB sam po sebi ne naplaćuje pohranu podataka jer je decentralizirana baza podataka koja koristi IPFS za pohranu i sinkronizaciju podataka. Ako korisnik koristi vlastite IPFS čvorove, nema dodatnih troškova osim troškova održavanja infrastrukture. Međutim, korištenje pružatelja usluga može stvoriti troškove ovisno o potrošnji podataka i širini pojasa. U većim sustavima cijena održavanja privatnih IPFS čvorova može porasti jer zahtijeva ulaganje u stabilne mreže i infrastrukturu. Stoga cijena pohrane u OrbitDB-u, iako primarno besplatna, je također i relativna i ovisi o razini decentralizacije koju korisnik želi postići te izborom infrastrukture, blockchain mreže i pružitelja usluga.

Konkretna cijena pohrane u sustavu IntegriDB trenutno nije jasno dostupna iz javno dostupnih izvora, no kao i ostali decentralizirani sustavi za pohranu podataka temeljeni na blockchainu, izravno je povezana s troškovima transakcija na blockchain mreži koja se koristi za pohranu podataka. Svaka transakcija koja uključuje unos ili ažuriranje podataka zahtijeva uplatu u vidu „gas fee“ naknada (u slučaju mreža poput Ethereuma) ili druge mrežne naknade u ovisnosti o blockchainu. IntegriDB je idealan za aplikacije koje ne zahtijevaju česta ažuriranja podataka jer je svaka interakcija s blockchainom povezana s dodatnim troškovima. Cijena pohrane u IntegriDB-u može značajno varirati ovisno o opterećenju mreže i volatilnosti kriptovalute povezane s mrežom. Primjerice, u vremenskim razdobljima visokog opterećenja blockchain mreže, naknade za transakcije mogu eksponencijalno rasti, što povećava ukupni trošak pohrane i može ograničiti financijsku održivost za aplikacije s velikim brojem unosa.

GroveDB je osmišljena kao baza podataka za decentralizirane aplikacije na Dash platformi, čiji ekosustav omogućuje učinkovitiju pohranu uz niže troškove transakcija nego u većim i opterećenijim blockchain mrežama. Cijena pohrane u GroveDB-u direktno ovisi o naknadama unutar Dash mreže, koje su općenito niže i stabilnije od većih javnih blockchain mreža. To omogućuje aplikacijama koje koriste GroveDB

da zadrže niske troškove transakcija pri pohrani podataka, što ih čini financijski održivima i za veće količine podataka ili učestalije ažuriranje.

Transakcije na Dash mreži naplaćuju naknadu koja se temelji na ukupnoj veličini bajtova transakcije. Naknada varira ovisno o potražnji za prostorom u blokovima — što je veća potražnja, to su više naknade. Minimalna naknada za prijenos transakcije preko mreže u studenom 2024. iznosi 0.00001 DASH-a, a prema [12] 1 DASH trenutno vrijedi oko 25 USD. Međutim, transakcije s minimalnim naknadama mogu se suočiti s dugim vremenom čekanja dok se ne nađe slobodan prostor u blokovima. Slika 5.1 prikazuje kako se transakcijska naknada za Dash mijenjala kroz povijest.



Slika 5.1 Transakcijska naknada za Dash [13]

Ukratko, pohrana podataka uključuje dinamične troškove koji se temelje na veličini transakcija i tržišnim uvjetima mreže na kojoj je sustav implementiran. Dakle, troškovi mogu varirati, a korisnici trebaju pažljivo razmotriti veličinu svojih podataka i potražnju na mreži kako bi optimizirali troškove. OrbitDB nudi ekonomičnost kroz decentralizaciju IPFS mreže, ali za visoku dostupnost može zahtijevati veće troškove za održavanje čvorova. IntegriDB je financijski isplativ za aplikacije koje ne zahtijevaju česte promjene podataka, dok cijena pohrane u GroveDB-u unutar

Dash platforme uvelike ovisi o veličini podataka koji se pohranjuju (broju bajtova) i trenutnoj zagušenosti mreže.

5.2 Vrsta podataka

Vrsta podataka koja se pohranjuje u decentralizirani sustav za pohranu podataka ovisi o njegovoj strukturi, sustavu pohrane i modelu podataka. Sustavi poput ObritDB, IntegriDB i GroveDB omogućuju pohranu različitih tipova podataka - od jednostavnih ključ-vrijednost parova do kompleksnih transakcijskih zapisa. Odabir baze podataka koja će se koristiti za neki projekt ovisi o zahtjevima aplikacije. Naprimjer, aplikacije za društvene mreže zahtijevaju često ažuriranje tekstualnih podataka, dok financijske ili vladine aplikacije traže nepromjenjivost i integritet podataka.

OrbitDB podržava pohranu različitih vrsta podataka: ključ-vrijednost, dokumente, zapise (engl. *logs*) i nizove zapisa koji se mogu kronološki pretraživati (engl. *feeds*). OrbitDB je idealan za aplikacije koje koriste tekstualne podatke i jednostavne strukture koje ne zahtijevaju visoku razinu konzistentnosti u stvarnom vremenu. Dokumentni sustav unutar OrbitDB-a olakšava pohranu strukturiranih zapisa, kao što su korisnički profili ili zapisi o aktivnostima, dok log strukture omogućavaju vođenje evidencije povijesti i bilježenje događaja. Osim toga, OrbitDB je optimiziran za aplikacije kojima su potrebni offline podaci koji se ažuriraju kada korisnici ponovno dobiju mrežnu povezanost, što ga čini pogodnim za društvene mreže. Iako podržava različite vrste podataka, OrbitDB nije dizajniran za pohranu velikih količina binarnih podataka ili visoko kompleksnih transakcija, što ga čini manje prikladnim za financijske aplikacije ili aplikacije s potrebama visoke sigurnosti.

IntegriDB podržava transakcijske podatke i SQL upite koji omogućavaju integritet i verifikaciju podataka. Ovaj sustav omogućuje pohranu kompleksnih podataka povezanih s transakcijama, osobito u kontekstu financijskih podataka, zbog čega je pogodna za sektore koji zahtijevaju nepromjenjivost podataka i visoku sigurnost. U IntegriDB-u, svaka transakcija ili promjena podataka pohranjuje se kao nepromjenjiv zapis na blockchainu, čime se osigurava transparentnost i otpornost na manipulaciju. Ovaj sustav podržava i složenije SQL upite za strukturiranu analizu podataka, što

omogućava visoku razinu fleksibilnosti u radu s kompleksnim podatkovnim strukturama. Često se koristi u područjima u kojima je potrebno često provjeravati identitet, transakcije ili integritet podataka. Iako nudi visoku razinu sigurnosti, IntegriDB nije idealan za aplikacije s velikim brojem učestalih i brzih transakcija.

GroveDB koristi ključ-vrijednost model pohrane i specijaliziran je za decentralizirane aplikacije u Dash platformi. Sustav je prilagođen za strukturirane podatke koji se lako pretražuju i brzo dohvaćaju. GroveDB podržava podatke specifične za kriptovalute i decentralizirane financijske aplikacije (DeFi), omogućujući efikasno pohranjivanje i pristup podacima vezanim uz račune korisnika, novčanike i povijest transakcija. Osim strukturiranih financijskih podataka, GroveDB omogućava decentraliziranu pohranu uz minimalne troškove, zbog čega je prikladan za platforme koje zahtijevaju visoku dostupnost podataka uz stabilne troškove. Najčešće se koristi u kriptovalutnim sustavima i aplikacijama za glasanje, gdje su brzi pristup i niska latencija ključni.

Vrste podataka čiju pohranu ove baze podržavaju definirane su prema potrebama aplikacija koje ih koriste. OrbitDB nudi fleksibilne opcije za aplikacije koje se oslanjaju na decentralizaciju i brzu replikaciju jednostavnijih podataka, dok je IntegriDB dizajniran za sigurnu i verifikabilnu pohranu transakcijskih podataka putem blockchaina. S druge strane, GroveDB se oslanja na efikasan pristup i pohranu strukturiranih podataka s niskim troškovima.

5.3 Način pohrane i pristupa

Način pohrane i pristupa podacima u distribuiranim bazama podataka ključan je kriterij koji određuje performanse, sigurnost i skalabilnost baze podataka. Ovaj kriterij obuhvaća tehnološku arhitekturu sustava te model pristupa podacima koji može biti direktan, replikacijski, temeljen na konsenzusu ili distribuiran kroz čvorove mreže. Različite metode pohrane podataka osiguravaju različite stupnjeve pouzdanosti, brzine pristupa i sigurnosti. Način pohrane ovisi o tehnologiji koja se koristi u sustavu, kao što je IPFS ili blockchain mreža, te o mogućnostima pristupa podacima putem P2P mreže ili specifičnih verifikacijskih sustava. Način pohrane također utječe na

učestalost ažuriranja, podršku za offline pristup, kao i na troškove zbog različitih metoda enkripcije, replikacije i arhiviranja podataka.

OrbitDB koristi IPFS za decentraliziranu pohranu podataka, gdje se podaci pohranjuju i repliciraju između čvorova u mreži. Podaci su strukturirani kao baze u obliku ključ-vrijednost parova, dokumenata i logova, čime se omogućava fleksibilna i brza pohrana. OrbitDB koristi CRDT-ove kako bi omogućio sinkronizaciju i automatsko rješavanje konflikata između različitih kopija baze podataka koje su distribuirane na više čvorova unutar mreže. CRDT je struktura podataka koja se replicira na više računala u mreži, bez potrebe za centraliziranim sustavom za upravljanje konfliktima. To znači da kad se čvorovi ponovno povežu, sve lokalne promjene se uspješno integriraju u zajedničku verziju baze. Pristup podacima u OrbitDB-u moguć je kroz P2P model koji osigurava visoku dostupnost i omogućava korisnicima da brzo sinkroniziraju promjene. [11]

Kod IntegriDB-a potrebna je verifikacija prilikom svakog unosa putem konsenzusnog modela, što podrazumijeva određene transakcijske troškove. No ipak je time osigurana visoka razina sigurnosti i integriteta. Pristup podacima u IntegriDB-u je sporiji zbog složenosti verifikacije, ali je također vrlo siguran. Način pohrane u ovom sustavu onemogućuje brze i česte izmjene podataka, što nije prikladno za aplikacije koje zahtijevaju brzinu i velik broj ažuriranja. Međutim, prednost IntegriDB-a je mogućnost pružanja transparentnosti i sigurnosti kroz nepromjenjivost podataka.

GroveDB implementira hibridni model pohrane na Dash platformi, koristeći kombinaciju strukture ključ-vrijednost s posebnim opcijama za decentralizirano spremanje i dohvaćanje podataka. GroveDB podržava visoko strukturirane podatke i omogućava brzi pristup. Način pohrane u GroveDB-u omogućava brze transakcije unutar ekosustava Dash platforme, a pristup podacima je optimiziran za Dash čvorove, što može ograničiti njegovu fleksibilnost u primjenama izvan ovog ekosustava. GroveDB je posebno koristan za aplikacije koje trebaju brze i česte izmjene podataka u sigurnom decentraliziranom okruženju.

Na temelju usporedbe načina pohrane i pristupa podacima, može se zaključiti da svaki od sustava ima specifične prednosti i mane prilagođene različitim potrebama i kontekstima uporabe. OrbitDB nudi fleksibilnost u radu s podacima kroz decentraliziranu pohranu na IPFS-u i P2P sinkronizaciju, što ga čini idealnim za aplikacije

kojima je potrebna autonomija korisnika mogućnost offline rada. IntegriDB je najpogodniji za aplikacije koje trebaju neporecivost i povijesnu točnost podataka. Iako je ovaj model izuzetno pouzdan, njegove performanse su niže zbog procesa verifikacije transakcija, čime se povećava sigurnost, ali smanjuje brzina pristupa. GroveDB balansira između sigurnosti i brzine pristupa koristeći hibridnu arhitekturu za decentraliziranu pohranu podataka unutar Dash platforme. Svaka od ovih baza podataka donosi različit pristup pohrani i pristupu podacima, a odabir odgovarajućeg sustava ovisi o prioritetima poput brzine, sigurnosti, konzistencije i fleksibilnosti.

5.4 Dostupnost dokumentacije i aktualnost održavanja

Dostupnost kvalitetne dokumentacije pomaže korisnicima da razumiju funkcionalnosti sustava i mogućnosti integracije. Dobra dokumentacija često podrazumijeva primjere i reference koje ubrzavaju proces integracije i razvoja. Osim dokumentacije, važan faktor u održavanju baze podataka je i zajednica korisnika. Aktivna zajednica često znači bržu podršku, više ažuriranja i popravke eventualnih kvarova. Redovito održavani sustavi jamče korisnicima sigurnost da će se baza podataka prilagođavati novim sigurnosnim i funkcionalnim zahtjevima, a to smanjuje potrebu za čestim migracijama.

OrbitDB nudi dokumentaciju koja pokriva osnove instalacije i primjere korištenja te GitHub repozitorij za brzi početak pod nazivom "OrbitDB Liftoff". Međutim, dokumentacija se može činiti nedovoljnom za naprednije postavke ili optimizaciju performansi. OrbitDB je projekt otvorenog koda, svatko može proširiti OrbitDB uključivanjem u projekt ili se uključiti u planiranje njegovog budućeg razvoja putem Matrix chata. U studenom 2024. godine OrbitDB ima 75 doprinositelja (engl. *contributors*), 569 forkova na GitHub-u, a posljednja verzija (v2.4.3) je objavljenja 1. studenog 2024.

IntegriDB nudi dokumentaciju u obliku istraživačkog rada koja pokriva kako je baza implementirana, koje napredne SQL mogućnosti podržava te optimizaciju performansi. Dokumentacija sadrži primjere, često postavljena pitanja te savjete za

integraciju u složena okruženja. Za razliku od OrbitDB-a, IntegriDB se ne čini kao redovito održavana baza podataka. Ažuriranja i nove verzije ne izlaze često i iza nje ne stoji velika zajednica korisnika ili aktivnih programera. Zbog toga bi korisnici mogli naići na poteškoće ako baza postane zastarjela ili ako se pojave problemi koje je potrebno brzo riješiti.

GroveDB ima razvijenu i lako dostupnu dokumentaciju na GitHub-u i službenoj stranici s uputama za instalaciju i konfiguraciju u različitim okruženjima. Osim osnovne dokumentacije, zajednica korisnika GroveDB-a osigurava podršku za napredne postavke i redovita ažuriranja pa je GroveDB dobar izbor za korisnike koji traže dugoročno održiv sustav. U studenom 2024. godine GroveDB ima 13 doprinositelja, 17 forkova na GitHub-u, a posljednja verzija (v2.1.0) je objavljena 4. listopada 2024.

Zaključno, OrbitDB ima dokumentaciju koja pokriva osnove instalacije i korištenja te GitHub repozitorij s primjerima, no može biti nedovoljna za naprednije postavke. Od svih sustava koji su uspoređeni u ovom radu, OrbitDB ima najveću popularnost, što je vidljivo kroz velik broj forkova i doprinositelja na GitHub-u. IntegriDB nudi detaljnu dokumentaciju u obliku istraživačkog rada, ali zbog nedostatka aktivnog razvoja i podrške zajednice korisnici mogu naići na poteškoće u održavanju. GroveDB nudi razvijenu dokumentaciju uz aktivno održavanje, što ga čini pogodnim za aplikacije koje zahtijevaju sigurnosne standarde i dugoročnu podršku. Koliko je aktivna zajednica korisnika pojedinog sustava i jesu li aktualno održavani, vidljivo je u Tablici 5.1.

Tablica 5.1 Aktualnost održavanja OrbitDB-a, IntegriDB-a i GroveDB-a

	Doprinositelji	**Zadnja verzija**	**Posljednje ažuriranje**
OrbitDB	75	v2.4.3	1.11.2024.
IntegriDB	2	Ne postoji službeno izdanje (engl. *release*); posljednja izmjena programskog koda 2015. godine	
GroveDB	13	v2.1.0	4.10.2024.

5.5 Lakoća instalacije i korištenja

Lakoća instalacije i korištenja nekog sustava jedan je od aspekata koji je ključan u projektima gdje je važno brzo postavljanje ili jednostavna integracija u postojeću infrastrukturu. Osim same instalacije, jednostavnost korištenja uključuje i intuitivnost sučelja te jednostavnost izvršavanja upita. Na lakoću korištenja utječe i dostupnost kvalitetne dokumentacije, koja pomaže korisniku da razumije funkcionalnosti, najbolje prakse i moguća rješenja za često postavljana pitanja.

OrbitDB

OrbitDB ima jednostavnu instalaciju pomoću Node.js okruženja, što ga čini povoljnim za JavaScript razvojne programere. Prvo, potrebno je instalirati Node.js, a zatim s *npm* (Node Package Manager) naredbom instalirati paket *@orbitdb/liftoff*. Ova instalacija se izvodi putem naredbe u terminalu:

```
npm install @orbitdb/liftoff
```

te se paket dodaje u *package.json* kao zavisnost (ukoliko je projekt već inicijaliziran sa *npm init*).

Potrebno je konfigurirati pakete *helia*, koji je implementacija IPFS-a u JavaScriptu, te *libp2p*, koji služi za sinkronizaciju baze podataka između peerova. Svi uvezeni paketi za izradu OrbitDB baze podataka prikazani su u Isječku koda 5.1.

```
1 import { createHelia } from 'helia'
2 import { createLibp2p } from 'libp2p'
3 import { createOrbitDB } from '@orbitdb/core'
4 import { DefaultLibp2pOptions } from './config/libp2p/index.js'
5 import { LevelBlockstore } from 'blockstore-level'
6 import { bitswap } from '@helia/block-brokers'
```

Isječak koda 5.1 Biblioteke i konfiguracije za OrbitDB

Inicijalizacija OrbitDB baze podataka zahtijeva konfiguraciju Libp2p, mrežnog sloja za komunikaciju između čvorova u peer-to-peer mreži. Nakon toga se definira

lokalno skladište blokova koristeći *LevelBlockstore*, koji sprema podatke u *./ipfs/-blocks* direktorij. Zatim se kreira OrbitDB instanca koja koristi Helia instancu za upravljanje IPFS protokolom. Helia kombinira Libp2p, *blockstore* i *bitswap* (implementacija blokovnog posrednika koji omogućava razjmenu podataka između čvorova) kako bi omogućila upravljanje podacima u IPFS mreži. Postupak je prikazan u Isječku koda 5.2.

```
1 async function initializeOrbitDB() {
2     const options = DefaultLibp2pOptions
3     const libp2p = await createLibp2p({ ...options })
4     const directory = '.'
5     const blockstore = new LevelBlockstore(`${directory}/ipfs/
          blocks`)
6     const ipfs = await createHelia({ libp2p, blockstore,
          blockBrokers: [bitswap()] })
7     const orbitdb = await createOrbitDB({ ipfs, directory })
8     return orbitdb
9 }
```

Isječak koda 5.2 Incijalizacija OrbitDB-a

Prije korištenja, potrebno je otvoriti bazu pomoću metode *open* (Isječak koda 5.3). Ukoliko baza sa zadanim imenom ne postoji, stvara se nova.

```
1 async function openDatabase(orbitdb, databaseName) {
2     const database = await orbitdb.open(databaseName);
3     return database;
4 }
```

Isječak koda 5.3 Otvaranje OrbitDB baze podataka

Metoda *add* koristi se za dodavanje novih podataka u bazu (Isječak koda 5.4). Pritom se koristi CRDT (struktura podataka koja omogućuje rad bez konflikata), što olakšava sinkronizaciju podataka među čvorovima u distribuiranoj mreži.

```
1 async function addDataToDatabase(database, data) {
2     await database.add(data);
```

```
3 }
```

Isječak koda 5.4 Dodavanje zapisa u OrbitDB bazu podataka

Dohvat svih zapisa iz baze podataka moguć je metodom *all*. Isječak koda 5.5 prikazuje dohvat i ispis svih zapisa.

```
1 async function printAllData(database) {
2     const allData = await database.all();
3     console.log(allData);
4 }
```

Isječak koda 5.5 Dohvaćanje svih zapisa iz OrbitDB baze podataka

Na kraju je potrebno zatvoriti bazu podataka. Metodom *stop* zaustavlja se rad baze podataka i IPFS instance. Na taj način se osigurava pravilno zatvaranje svih otvorenih resursa, uključujući skladište podataka i mrežne veze. Postupak zatvaranja prikazan je u Isječku koda 5.6.

```
1 async function closeOrbitDB(orbitdb) {
2     await orbitdb.stop()
3     await orbitdb.ipfs.stop()
4     await orbitdb.ipfs.blockstore.unwrap().unwrap().child.db.
          close()
5 }
```

Isječak koda 5.6 Zatvaranje OrbitDB baze podataka

IntegriDB

Instalacija IntegriDB-a počinje preuzimanjem repozitorija s GitHub stranice:

```
git clone https://github.com/integridb/Code.git
```

Nakon preuzimanja koda, potrebno je konfigurirati server na kojem će baza raditi. Ovo uključuje instalaciju potrebnih softverskih alata za pokretanje SQL okruženja te dodatne prilagodbe ovisno o specifičnim zahtjevima aplikacije. IntegriDB se oslanja na sljedeće biblioteke:

- **MySQL Server** kao SQL poslužitelj baze podataka
- **MySQL Client C++ biblioteke** kao SQL API
- **OpenSSL** za enkripciju i *hashing*
- **ate-pairing**
- **xbyak**
- **NTL** i **GMP** za teoriju brojeva

Za kreiranje testnih tablica, unutar *Code/* direktorija potrebno je pokrenuti naredbe:

```
make create
./createtables
```

Zatim, za pokretanje glavnog koda koriste se sljedeće naredbe:

```
make main
./main
```

Međutim, posljednje ažuriranje ovog projekta bilo je 2015. godine, što implicira da koristi zastarjele verzije gore navedenih biblioteka. Zbog toga je tijekom istraživanja i pokušaja instalacije došlo do brojnih tehničkih poteškoća što je rezultiralo nemogućnošću pokretanja ove baze podataka za testiranje. Ovi izazovi ukazuju na ograničenu praktičnost IntegriDB-a u suvremenim uvjetima te naglašavaju važnost kontinuiranog održavanja i prilagodbe novim standardima.

GroveDB

GroveDB zahtijeva poznavanje i korištenje programskog jezika Rust. Za preuzimanje i instalaciju Rust-a potrebno je pokrenuti sljedeću naredbu:

```
curl --proto '=https' --tlsv1.2 -sSf https://sh.rustup.rs | sh
```

te je onda moguće pomoću naredbe *cargo* pokrenuti GroveDB kod:

```
git clone https://github.com/masternodes/grovedb.git
cd grovedb
cargo build --release
cargo run --bin <naziv datoteke>
```

GroveDB koristi funkciju *open* za otvaranje postojeće instance GroveDB-a na zadanoj putanji u datotečnom sustavu. Ako instanca na toj putanji ne postoji, funkcija će kreirati novu instancu s praznim korijenskim stablom. Jedini argument funkcije *open* je putanja u datotečnom sustavu. Inicijalizacija i otvaranje GroveDB baze podataka prikazani su u Isječku koda 5.7.

```
let path = String::from("../storage");
let database = GroveDb::open(&path).unwrap();
```

Isječak koda 5.7 Otvaranje GroveDB baze podataka

Za unos ključ-vrijednost stavki u bazu podataka, GroveDB koristi funkciju *insert*. Funkcija prima pet argumenata: put do podstabla gdje treba umetnuti zapis ("[ ]" za korijensko stablo), ključ, vrijednost, opcije umetanja te transakcija u koju treba uključiti operaciju umetanja. Primjer umetanja zapisa u korijensko stablo prikazan je u Isječku koda 5.8.

```
let key = b"key";
let val = b"value";
database.insert([], key, Element::Item(val.to_vec(), None),
    None, None)
    .unwrap()
    .expect("Successfully inserted");
```

Isječak koda 5.8 Umetanje zapisa ključ-vrijednost u GroveDB bazu podataka

GroveDB koristi funkciju *delete* za brisanje ključeva-vrijednosti iz pohrane. Potrebna su četiri argumenta: put do podstabla gdje se nalazi ključ ("[ ]" za korijensko stablo), ključ koji se briše, opcije brisanja te transakcija u koju treba uključiti operaciju brisanja. Primjer brisanja ključa prikazan je u Isječku koda 5.9.

```
database.delete([], key1, None, None)
    .unwrap()
    .expect("successfully deleted");
```

Isječak koda 5.9 Brisanje ključa iz GroveDB baze podataka

Za dohvaćanje jednog zapisa koristi se funkcija *get*, kako je prikazano u Isječku koda 5.10. Funkcija *get* omogućа samo dohvaćanje jedne stavke, ključ mora biti naveden, a kriptografski dokazi nisu podržani. S druge strane, upiti (jednostavni ili kompleksni) mogu vratiti više vrijednosti odjednom, ključevi se ne moraju dati, a rezultati upita mogu se kriptografski dokazati.

```
let result = database.get([], key, None)
    .unwrap();
```

Isječak koda 5.10 Dohvaćanje zapisa iz GroveDB baze podataka

5.6 Skalabilnost

Skalabilnost baze podataka je karakteristika sustava koja se odnosi na njegovu sposobnost prilagođavanja rastućim potrebama za obradom podataka i povećanom radnom opterećenju. Sustav koji se dobro skalira može održati ili čak povećati razinu performansi ili učinkovitosti, čak i kad se povećavaju operativni zahtjevi. Skalabilnost može biti horizontalna (dodavanje novih čvorova ili instanci) ili vertikalna (dodavanje resursa kao što su procesorska snaga ili memorija u postojećim čvorovima). [14]

Kod distribuiranih baza podataka, skalabilnost često ovisi o strukturi i načinu replikacije podataka, arhitekturi mreže (P2P ili centralizirano) te načinima upravljanja podataka. Sustavi s visokom razinom skalabilnosti mogu učinkovito prilagoditi rastuće potrebe za obradom podataka, a time i poboljšati performanse i dostupnost usluge.

OrbitDB je distribuirana baza podataka koja koristi IPFS i P2P mrežu pa pruža visoku razinu horizontalne skalabilnosti. Ova struktura omogućava svakom čvoru da samostalno pohranjuje podatke i dijeli ih s drugim čvorovima, što omogućuje efikasno skaliranje u smislu broja korisnika. Međutim, može doći do izazova kod skaliranja s

velikim količinama podataka jer IPFS može imati veću latenciju u odnosu na klasične sustave s bržim pristupom podacima.

Testiranje skalabilnosti OrbitDB-a provedeno je s ciljem mjerenja vremena unosa različitog broja zapisa u bazu. Za svaki od brojeva zapisa (1, 100, 1000, 5000 i 10000 zapisa) mjerenja su ponovljena deset puta, a zatim je izračunata prosječna vrijednost za svaku kategoriju. Pritom su korišteni zapisi veličine 1 KB. Vrijeme unosa izraženo je u sekundama. Rezultati su prikazani u Tablici 5.2.

Tablica 5.2 Rezultati testiranja skalabilnosti OrbitDB-a

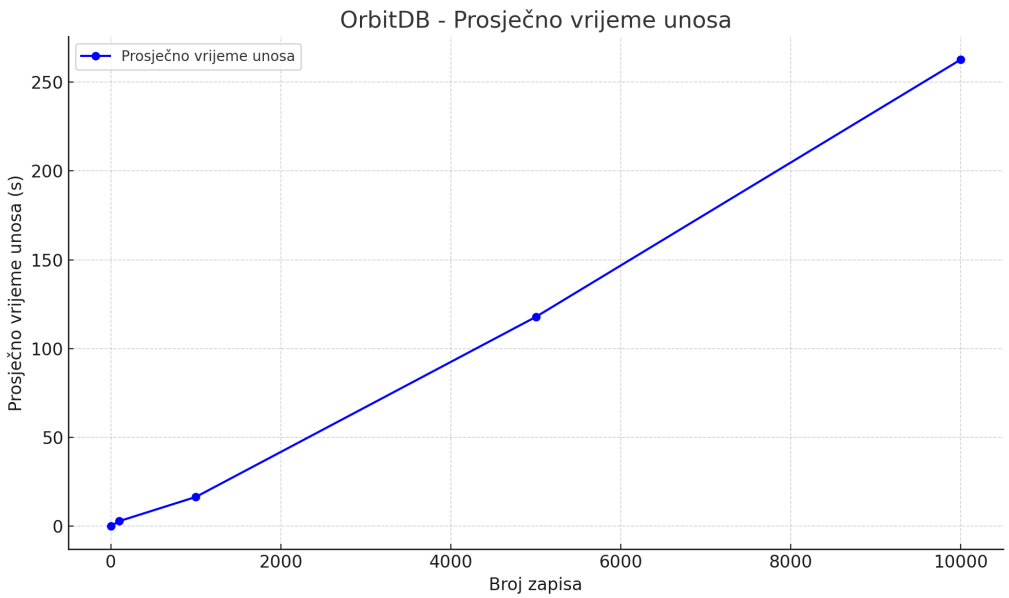
Broj zapisa	**Prosječno vrijeme unosa**
1 zapis	0,09 s
100 zapisa	2,8 s
1000 zapisa	16,41 s
5000 zapisa	117,81 s
10000 zapisa	262,75 s

Slika 5.2 prikazuje graf s prosječnim vremenom unosa u funkciji broja zapisa.

Testiranje je pokazalo da OrbitDB može rukovati različitim količinama podataka, no vrijeme unosa značajno raste s povećanjem broja zapisa.

U svrhu evaluacije skalabilnosti IntegrDB sustava provedeno je testiranje izvedbe JOIN upita na dvije tablice jednake veličine. Obje tablice sadržavale su po 10 stupaca, dok se broj redaka n mijenjao u rasponu od 100 do 100.000. Rezultati i analiza skalabilnosti i performansi IntegriDB-a izvedeni su iz rada [8]. Eksperiment je proveden na Amazon EC2 instanci s 16 GB RAM memorije i operacijskim sustavom Linux. Za svaku točku podataka prikupljeno je 10 mjerenja, a u Tablici 5.3 su prikazane prosječne vrijednosti. Eksperimentalni rezultati ovog testiranja pokazuju da vrijeme izračuna dokaza (engl. *prover time*) raste linearno s brojem redaka, dok vrijeme verifikacije ostaje gotovo konstantno, bez obzira na veličinu tablice.

Što se tiče GroveDB-a, Dash platforma omogućuje brzu horizontalnu skalabilnost kroz mrežu *masternodeova*, koji služe za pružanje naprednih usluga i upravljanje blockchainom te osiguravaju raspodjelu resursa i efikasniju obradu transakcija. GroveDB



Slika 5.2 Prosječno vrijeme unosa u OrbitDB u ovisnosti o broju zapisa

Tablica 5.3 Rezultati testiranja skalabilnosti IntegriDB-a [8]

Broj redaka	**Vrijeme izračuna dokaza**	**Veličina dokaza**	**Vrijeme verifikacije**
100	0,041 s	11,97 KB	40,7 ms
1000	1,38 s	16,77 KB	45,2 ms
10000	15,7 s	23,17 KB	45,3 ms
100000	168 s	27,97 KB	45,4 ms

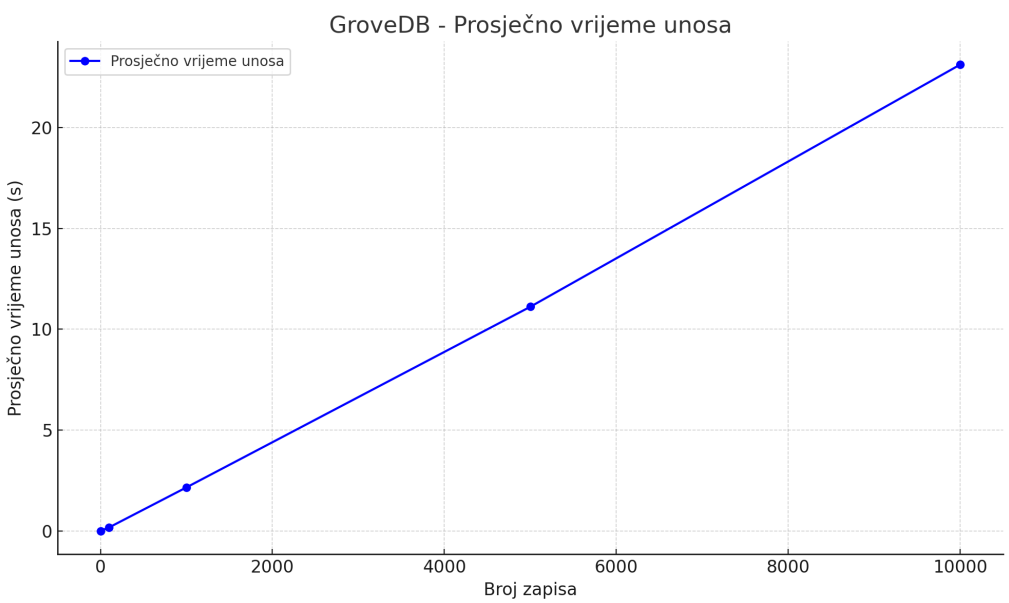
koristi strukturirani pristup koji omogućava brzu replikaciju i preuzimanje podataka, čak i pri velikom opterećenju mreže.

Testiranje skalabilnosti za GroveDB je, kao i kod OrbitDB-a, provedeno za različite količine zapisa (1, 100, 1000, 5000 i 10000 zapisa) veličine 1 KB. Vrijeme unosa je izraženo u sekundama, a prosječne vrijednosti su prikazane u Tablici 5.4. Testiranje skalabilnosti i performansi OrbitDB-a i GroveDB-a provedeno je na računalu s 4 GB RAM memorije i operacijskim sustavom Linux.

Tablica 5.4 Rezultati testiranja skalabilnosti GroveDB-a

Broj zapisa	**Prosječno vrijeme unosa**
1 zapis	1,878 ms
100 zapisa	174,133 ms
1000 zapisa	2,159 s
5000 zapisa	11,115 s
10000 zapisa	23,121 s

Na grafu (Slika 5.3) se vidi da prosječno vrijeme unosa raste linearno u odnosu na količinu unesenih zapisa kao i kod OrbiDB, međutim vrijeme potrebno za unos je i do 10 puta manje u odnosu na OrbitDB u slučaju velikog broja zapisa.



Slika 5.3 Prosječno vrijeme unosa u GroveDB u ovisnosti o broju zapisa

5.7 Sigurnost i pouzdanost

Pouzdanost decentraliziranog sustava za pohranu podataka se odnosi na sposobnost sustava da omogući kontinuiran, stabilan i točan pristup pohranjenim podacima, dok sigurnost obuhvaća zaštitu podataka od neovlaštenog pristupa te zaštitu od gubitka ili oštećenja podataka. Ovaj kriterij ovisi o nekoliko faktora, uključujući način autentifikacije korisnika, metode enkripcije, mehanizme za oporavak, dostupnost podataka, kao i implementaciju sigurnosnih mjera koje čuvaju integritet podataka. Pouzdana baza podataka smanjuje rizik od prekida rada aplikacije, osigurava postojanost podataka te time omogućuje bazama podataka da zadovolje sigurnosne standarde.

OrbitDB koristi logičku replikaciju podataka i mehanizme za automatsku sinkronizaciju, čime je osigurana visoka konzistentnost podataka kroz mrežu. Iako ova arhitektura povećava dostupnost podataka, OrbitDB ima ograničenja u odnosu na transakcijsku pouzdanost zbog P2P pristupa. Sigurnosni model se temelji na digitalnim potpisima koji osiguravaju autentičnost podataka. Iako ova arhitektura omogućuje osnovnu sigurnost, OrbitDB se oslanja na vanjske alate za enkripciju. U aplikacijama koje ne zahtijevaju stalnu transakcijsku točnost, OrbitDB pruža zadovoljavajuću razinu pouzdanosti.

IntegriDB stavlja poseban naglasak na integritet i točnost podataka te osigurava visoku razinu sigurnosti. Ovaj sustav primjenjuje napredne sigurnosne metode, uključujući enkripciju podataka i kontrolu pristupa. Za razliku od OrbitDB-a, IntegriDB osigurava snažnu transakcijsku konzistentnost, sprječavajući gubitak podataka ili neuspješne transakcije. Ovaj sustav je prikladniji za aplikacije gdje je ključno zadržati visoku razinu točnosti podataka, čak i u slučaju neočekivanih kvarova sustava, pogotovo za aplikacije koje upravljaju povjerljivim i visoko osjetljivim informacijama, kao što su financijski i zdravstveni sustavi.

GroveDB također ima snažan sigurnosni model koji koristi Merkle-AVL stabla za očuvanje integriteta podataka i sprječavanje manipulacije. GroveDB implementira različite sigurnosne mehanizme, uključujući enkripciju podataka te provjere autentičnosti za sprječavanje neovlaštenih pristupa. Osim toga, podržava verzioniranje podataka, što omogućuje praćenje i reviziju promjena te oporavak podataka u slučaju neovlaštenih izmjena ili kvara. Zbog ovih sigurnosnih značajki, GroveDB je

pouzdano rješenje za aplikacije kojima je potrebna visoka razina sigurnosti i transparentnosti podataka.

Sve tri baze podataka imaju značajke koje osiguravaju pouzdanost i primjenjuju specifične sigurnosne metode ovisno o svojoj strukturi i namjeni. OrbitDB je bolji za distribuirane aplikacije s manjim naglaskom na transakcijsku konzistentnost, IntegriDB osigurava stabilne transakcijske procese, dok GroveDB nudi pouzdane sigurnosne mjere s dodatnom transparentnošću kroz verzioniranje i praćenje podataka.

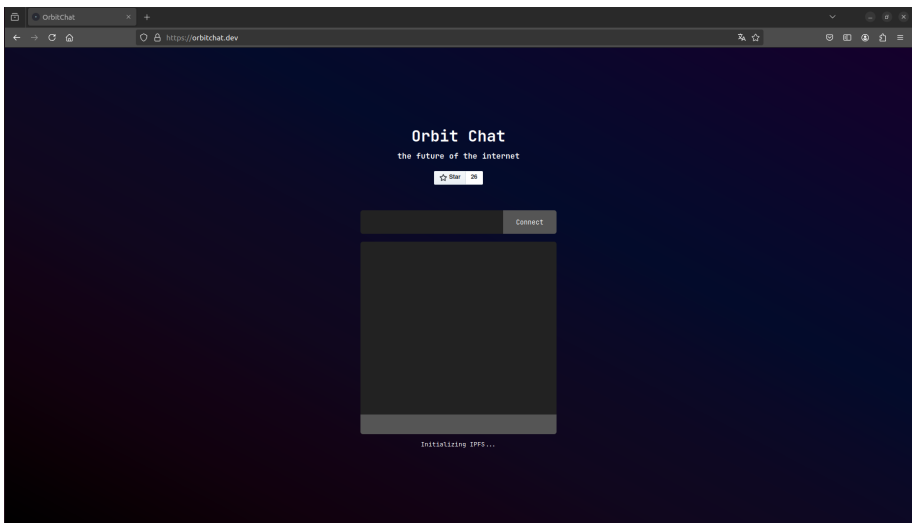
5.8 Primjena

Primjena decentraliziranih sustava za pohranu podataka ovisi o njihovim tehničkim karakteristikama, arhitekturi i specifičnim potrebama korisnika. Distribuirane baze podataka dizajnirane su za specifične vrste distribuiranih aplikacija, gdje su decentralizacija, sigurnost, skalabilnost i brzina pristupa ključni faktori. Razlike u načinu pohrane, vrsti podataka koja se pohranjuje i podršci za različite tipove transakcija određuju koja baza najbolje odgovara kojim scenarijima. Distribuirane baze podataka često se koriste u aplikacijama kojima je potrebna visoka otpornost na kvarove i sigurna pohrana podataka, što je presudno u sektorima poput financijskih usluga, društvenih mreža i IoT-a. Osim tehničkih karakteristika, primjena ovih baza također ovisi o potrebnoj privatnosti podataka te učestalosti ažuriranja.

OrbitDB je razvijen kao baza podataka za decentralizirane aplikacije koje zahtijevaju brzo repliciranje podataka i visok stupanj skalabilnosti, uz minimalne troškove infrastrukture. OrbitDB koristi IPFS, koji omogućuje decentralizirano pohranjivanje podataka i distribuiranu dostupnost. Ova baza podataka je pogodna za aplikacije u kojima korisnici trebaju lokalno pohranjene kopije podataka koje se sinkroniziraju u stvarnom vremenu s ostalim korisnicima. Najčešće se koristi u aplikacijama kao što su decentralizirane društvene mreže, platforme za dijeljenje sadržaja i blockchain aplikacije. OrbitDB također omogućuje da korisnici pristupe podacima bez potrebe za konstantnim povezivanjem sa poslužiteljem, što ga čini prikladnim za projekte s ograničenom dostupnošću mreže ili povećanim zahtjevima za privatnost podataka. Međutim, budući da OrbitDB nije dizajniran za vrlo velike količine podataka ili tran-

sakcije koje zahtijevaju strogo jamstvo konzistentnosti, nije idealan za financijske ili slične aplikacije.

Jedan primjer aplikacije koja koristi OrbitDB je Orbit Chat (Slika 5.4), decentralizirana aplikacija za chat koja koristi OrbitDB za pohranu i sinkronizaciju poruka između različitih korisničkih čvorova. Ova aplikacija omogućuje korisnicima da šalju poruke koje se zatim šire kroz sve čvorove u razgovoru, pritom svi sudionici mogu istovremeno vidjeti ažuriranja dok se podaci sinkroniziraju putem IPFS-a.



Slika 5.4 Primjer aplikacije koja koristi OrbitDB: Orbit Chat

IntegriDB je usmjeren na aplikacije koje zahtijevaju visoku razinu sigurnosti i neporecivost podataka. Kao baza podataka temeljena na blockchainu, IntegriDB koristi prednosti nepromjenjivosti i transparentnosti blockchain mreže kako bi osigurao da svi zapisi podataka ostaju trajno pohranjeni i dostupni za verifikaciju. Ovaj sustav je posebno prikladan za sektore u kojima je ključno povjerenje između sudionika, kao što su financijski sustavi i pravne aplikacije. IntegriDB se često koristi u aplikacijama gdje se podaci rijetko ažuriraju, ali je važno zadržati zapis o svim promjenama radi kasnijeg pregleda ili revizije. Prikladan je za aplikacije u kojima su transakcijski podaci osjetljivi na manipulaciju jer svaki unos postaje dio blockchaina, čime se osigurava visok stupanj sigurnosti i otpornosti na izmjene. Međutim, zbog troškova transakcija na blockchain mreži i vremenskih ograničenja obrade, IntegriDB

nije najprikladniji za aplikacije s velikim brojem brzih i učestalih transakcija.

GroveDB je dizajniran za aplikacije unutar Dash platforme s naglaskom na decentralizirane aplikacije koje zahtijevaju stabilne troškove te brzinu pristupa podacima. GroveDB podržava visoko strukturirane podatke putem ključ-vrijednost modela, što ga čini pogodnim za aplikacije koje koriste brze pretrage i visoku dostupnost podataka. Ova baza podataka je posebno optimizirana za aplikacije koje se temelje na ekonomiji kriptovaluta, decentraliziranim financijama (DeFi) i sustavima glasanja. GroveDB podržava decentralizirane aplikacije koje zahtijevaju efikasan sustav s niskim troškovima za upravljanje velikim količinama podataka i učestalim ažuriranjima. Osim toga, integracija unutar Dash ekosustava čini GroveDB jednostavnim za implementaciju u aplikacije koje već koriste Dash platformu i kojima su potrebni predvidivi i stabilni troškovi. Ova baza podataka nije idealna za aplikacije izvan Dash ekosustava zbog potencijalne složenosti integracije i ovisnosti o specifičnim funkcijama platforme.

Ukratko, OrbitDB, IntegriDB i GroveDB se razlikuju po mogućim scenarijima primjene, ovisno o njihovim tehničkim karakteristikama. OrbitDB je prikladan za aplikacije kojima je brzo repliciranje podataka ključno, dok je IntegriDB optimalan izbor za aplikacije koje zahtijevaju nepromjenjivost podataka i visoku razinu sigurnosti. GroveDB je specijaliziran za aplikacije na Dash platformi koje zahtijevaju predvidive troškove i visoku efikasnost u pristupu podacima.

5.9 Zaključak usporedbe

U Tablici 5.5 prikazane su ključne karakteristike svakog decentraliziranog sustava za pohranu podataka kojeg analiziramo u ovom radu. Pregled ključnih karakteristika u odnosu na određene kriterije olakšava usporedbu i razumijevanje ključnih razlika između njih.

Tablica 5.5 Usporedba OrbitDB, IntegriDB i GroveDB prema osnovnim kriterijima

Kriterij	**OrbitDB**	**IntegriDB**	**GroveDB**
Cijena pohrane	Bez direktnih troškova, ovisi o IPFS-u	Ovisi o mrežnoj naknadi odabranog blockchaina	Ovisi o mrežnim troškovima na Dash platformi, koji su općenito niži u odnosu na druge javne blockchain mreže)
Vrsta podataka	Ključ-vrijednost, dokumenti, logovi feedovi	Podaci koji zahtijevaju verifikaciju (npr. transakcije)	Ključ-vrijednost model pohrane, strukturalni podaci za brzi pristup
Način pohrane i pristupa	IPFS, CRDT, P2P	Podržava SQL upite za pohranu i pristup, omogućava verifikaciju podataka	Pohrana na Dash platformi (brzo indeksiranje)
Dostupnost dokumentacije i aktualnost održavanja	Dobro dokumentirano na GitHub-u i službenoj stranici, redovito održavano	Dokumentacija dostupna u obliku istraživačkog rada, ali ograničena, posljednji put ažurirano 2015.	Dobro dokumentirano na Dash platformi i službenim stranicama
Lakoća instalacije i korištenja	Srednja, zahtijeva poznavanje IPFS-a i JavaScript-a	Zahtjevna, koristi zastarjele verzije biblioteka	Srednja, zahtijeva poznavanje Rust-a

Kriterij	**OrbitDB**	**IntegriDB**	**GroveDB**
Skalabilnost	Srednja, ograničena IPFS-om i brojem čvorova	Vrijeme verifikacije ostaje konstantno povećanjem veličine tablice	Visoka, zbog korištenja Dash masternode sustava
Sigurnost i pouzdanost	Visoka, oslanja se na IPFS infrastrukturu	Visoka, osigurava transakcijsku konzistentnost	Visoka, koristi kriptografsku zaštitu u sklopu Dash platforme
Primjena	Decentralizirane aplikacije (dApps), aplikacije koje zahtijevaju offline rad	Decentralizirane baze podataka za poslovne aplikacije	Decentralizirane aplikacije na Dash platformi

Poglavlje 6

Zaključak

Decentralizirani sustavi za pohranu podataka donose značajne promjene u načinu na koji pohranjujemo, dijelimo i pristupamo podacima. Ovaj rad analizirao je tri specifična sustava – OrbitDB, IntegriDB i GroveDB – te pružio uvid u njihove prednosti, nedostatke i mogućnosti primjene u različitim kontekstima. Kroz istraživanje je postalo jasno kako decentralizacija, iako zahtijeva složenije tehnološke pristupe, pruža brojne pogodnosti u vidu sigurnosti, skalabilnosti i otpornosti na tehničke kvarove.

Jedan od glavnih zaključaka jest da decentralizacija omogućuje veću sigurnost i pouzdanost podataka eliminiranjem središnjih točaka kvarova i oslanjanjem na distribuirane mreže. Tehnologije poput blockchaina i IPFS-a osiguravaju integritet i dostupnost podataka čak i u nepovoljnim uvjetima. Sustavi poput OrbitDB-a jednostavni su za implementaciju u decentraliziranim aplikacijama, dok IntegriDB osigurava visoku razinu provjerljivosti i sigurnosti podataka, što je ključna značajka u sektorima poput financija i pravnih sustava. S druge strane, GroveDB je prilagođen aplikacijama kojima je potrebna efikasna pohrana i brzi pristup, osobito unutar Dash platforme.

Ipak, treba uzeti u obzir i izazove koji prate decentralizirane sustave. Implementacija može biti složena, a održavanje zahtijeva specijalizirano znanje. Također, troškovi skalabilnosti i interoperabilnosti s postojećim sustavima ostaju područja koja zahtijevaju daljnji razvoj.

U budućnosti, napredak tehnologija poput održivog blockchaina, optimizacije

IPFS-a i kvantne kriptografije mogao bi dodatno smanjiti postojeće prepreke i omogućiti širu primjenu decentraliziranih sustava. Njihova integracija s pametnim uređajima, umjetnom inteligencijom i IoT-om otvara nove mogućnosti za inovacije, dok daljnja istraživanja mogu pridonijeti razvoju efikasnijih i ekonomičnijih rješenja.

Zaključno, decentralizirani sustavi za pohranu podataka nisu samo alternativa tradicionalnim metodama, već značajan korak naprijed u razvoju digitalne infrastrukture. Rezultati ovog rada pružaju polaznu osnovu za razumijevanje njihovih mogućnosti i izazova te poziv na daljnja istraživanja i prilagodbe kako bi se u potpunosti iskoristio njihov potencijal u raznim industrijama.

Bibliografija

[1] Wikipedia: *Blockchain*, s Interneta: `https://en.wikipedia.org/wiki/Blockchain`, kolovoz 2024.

[2] Orešković, D: *Vrste konsenzus algoritama*, s Interneta: `https://crobitcoin.com/vrste-konsenzusa-na-blockchainu-proof-of-work-vs-proof-of-stake`, 9. siječnja 2020.

[3] BasuMallick, C.: *What Are Smart Contracts: Types, Benefits, and Tools*, s Interneta: `https://www.spiceworks.com/tech/innovation/articles/what-are-smart-contracts/`, 20. studenog 2020.

[4] Wegrzyn, K.; Wang, E.: *Types of Blockchain: Public, Private, or Something in Between*, s Interneta: `https://www.foley.com/insights/publications/2021/08/types-of-blockchain-public-private-between/`, 19. kolovoza 2021.

[5] Norman, D.: *A Practical Explainer for IPFS Gateways - Part 1*, s Interneta:`https://blog.ipfs.tech/2022-06-09-practical-explainer-ipfs-gateways-1/`, 9. lipnja 2022.

[6] Wikipedia: *InterPlanetary File System*, s Interneta: `https://en.wikipedia.org/wiki/InterPlanetary_File_System`, kolovoz 2024.

[7] OrbitDB - Home: *What is OrbitDB*, s Interneta: `https://orbitdb.org/`, kolovoz 2024.

[8] Katz, J.; Papamanthou, C.; Zhang, Y.: *IntegriDB: Verifiable SQL for Outsourced Databases*, s Interneta: `http://integridb.github.io/IntegriDB.pdf`, 2015.

[9] GroveDB: *The first database to support cryptographic proofs for complex queries*, s Interneta: `https://www.grovedb.org/`, 2023.

[10] Wikipedia: *ACID (računarstvo)*, s Interneta: `https://hr.wikipedia.org/wiki/ACID_(ra%C4%8Dunarstvo)`, siječanj 2022.

[11] Wikipedia: *Conflict-free replicated data type*, s Interneta: `https://en.wikipedia.org/wiki/Conflict-free_replicated_data_type`, studeni 2024.

[12] My Dash Wallet: *Transaction Fees*, s Interneta: `https://old.mydashwallet.org/AboutTransactionFees`, studeni 2024.

[13] Blockchair: *Dash charts*, s Interneta: `https://blockchair.com/dash/charts`, studeni 2024.

[14] Hoogenraad, W.: *Skalabilnost kao softverski zahtjev, značenje i definicija*, s Interneta: `https://hr.itpedia.nl/2021/07/20/schaalbaarheid-als-software-requirement-betekenis-en-definitie/`, srpanj 2021.

Sažetak

U ovom radu analiziraju se decentralizirani sustavi za pohranu podataka temeljeni na blockchain tehnologiji. Kao primjeri su odabrana tri postojeća rješenja: OrbitDB, IntegriDB i GroveDB, čija su svojstva i arhitektura detaljno opisani. Fokus istraživanja bio je na usporedbi ovih sustava prema odabranim kriterijima, uključujući cijenu pohrane, skalabilnost, vrstu podataka koja se pohranjuje, primjenu, sigurnost, pouzdanost te lakoću instalacije i korištenja. Analiza je otkrila prednosti i nedostatke svakog rješenja, omogućujući bolji uvid u njihovu prikladnost za različite primjene.

Ključne riječi — **blockchain, IPFS, decentralizacija**

Abstract

This paper analyzes decentralized data storage systems based on blockchain technology. Three existing solutions were selected as examples: OrbitDB, IntegriDB, and GroveDB, whose features and architecture are described in detail. The research focused on comparing these systems based on selected criteria, including storage cost, scalability, type of data stored, applications, security, reliability, and ease of installation and use. The analysis highlighted the advantages and disadvantages of each solution, providing insights into their suitability for different use cases.

Keywords — **blockchain, IPFS, decentralization**