

Native Website Hosting on Blockchain

as long you have index.html in your prod build...

Apoorv Gupta

<https://twitter.com/Apoorv2204>

The Current State.

The major focus of decentralisation using blockchain technology was on the backend, or infrastructure side of things, especially with smart contracts and decentralised applications (DApps) on blockchains like polygon,eth,atom,etc.

The frontend of many DApps and blockchain platforms is still centralized. For a truly decentralised application, both the backend and frontend should be decentralised.

The Proposal

The objective of this is an attempt to host frontend websites directly from the blockchain nodes. Using the existing power of the Comos-SDK to create a sovereign blockchain,allowing to create and use custom tx and messages, enabling the seamless hosting of websites built intrinsically on the blockchain network.

- Create a Custom Blockchain that can host website from transaction

How it will be when its built?

01

Compile and Build

Begin by compiling and building your front-end project using your preferred tools or frameworks

02

Create a Transaction

After successfully deploying or transacting on the blockchain, note down the transaction hash.

03

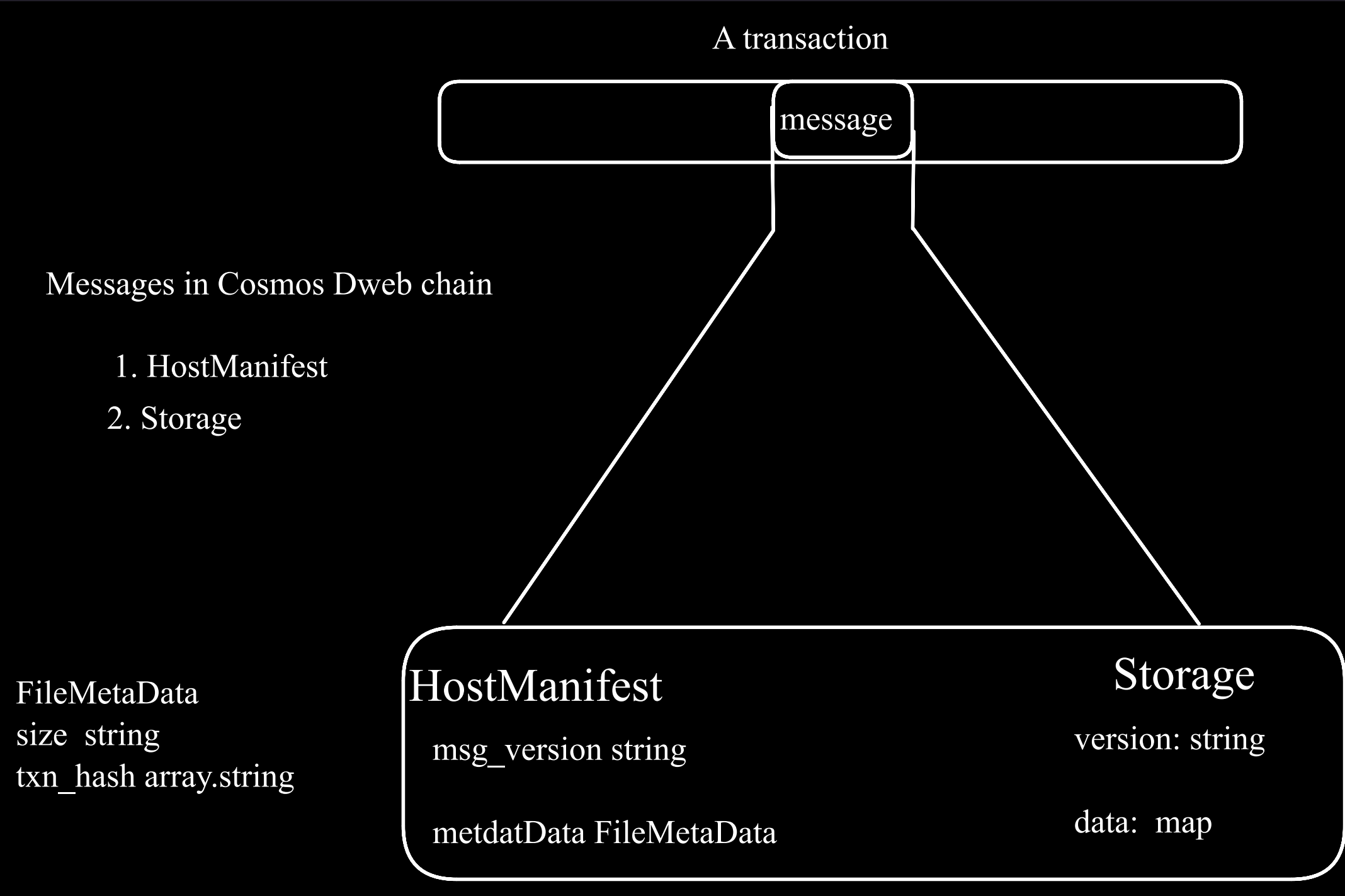
Access the Deployed Website:

Navigate to the node's endpoint followed by the transaction hash: `{node_endpoint}/{tx_hash}`. This URL should display the website hosted directly from the blockchain node.

Technicalities

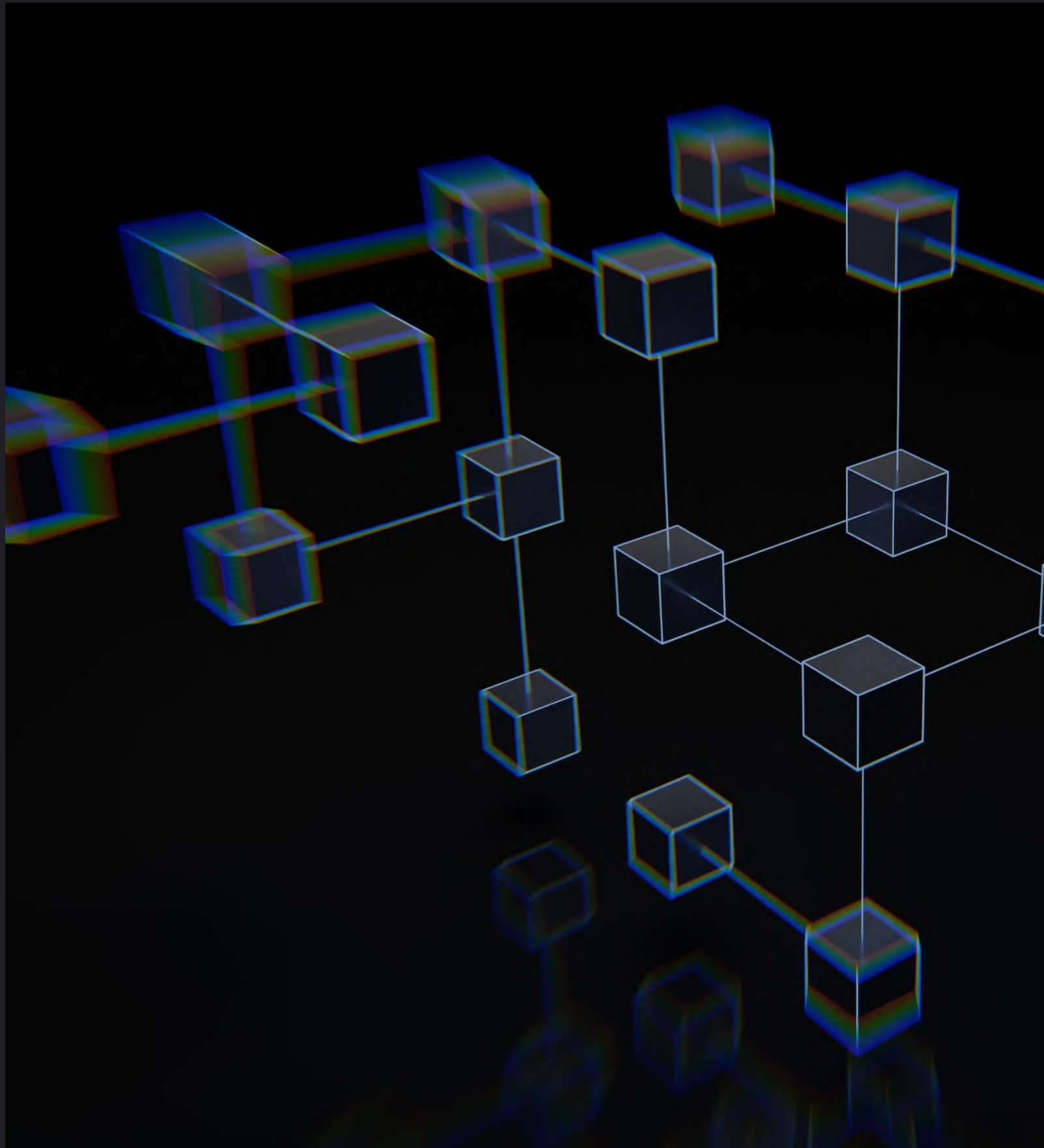
In Simple term: Put website information in the transaction itself. And host the website from the stored transaction

- **Host Manifest:**
A table of contents for your website, showing where each file is.
- **Storage Message:**
Holds the website's files in a encoded form.
- **Access via Tx Hash:**
When someone wants to visit your website, using tx hash, The blockchain decodes the tx and serves the website.



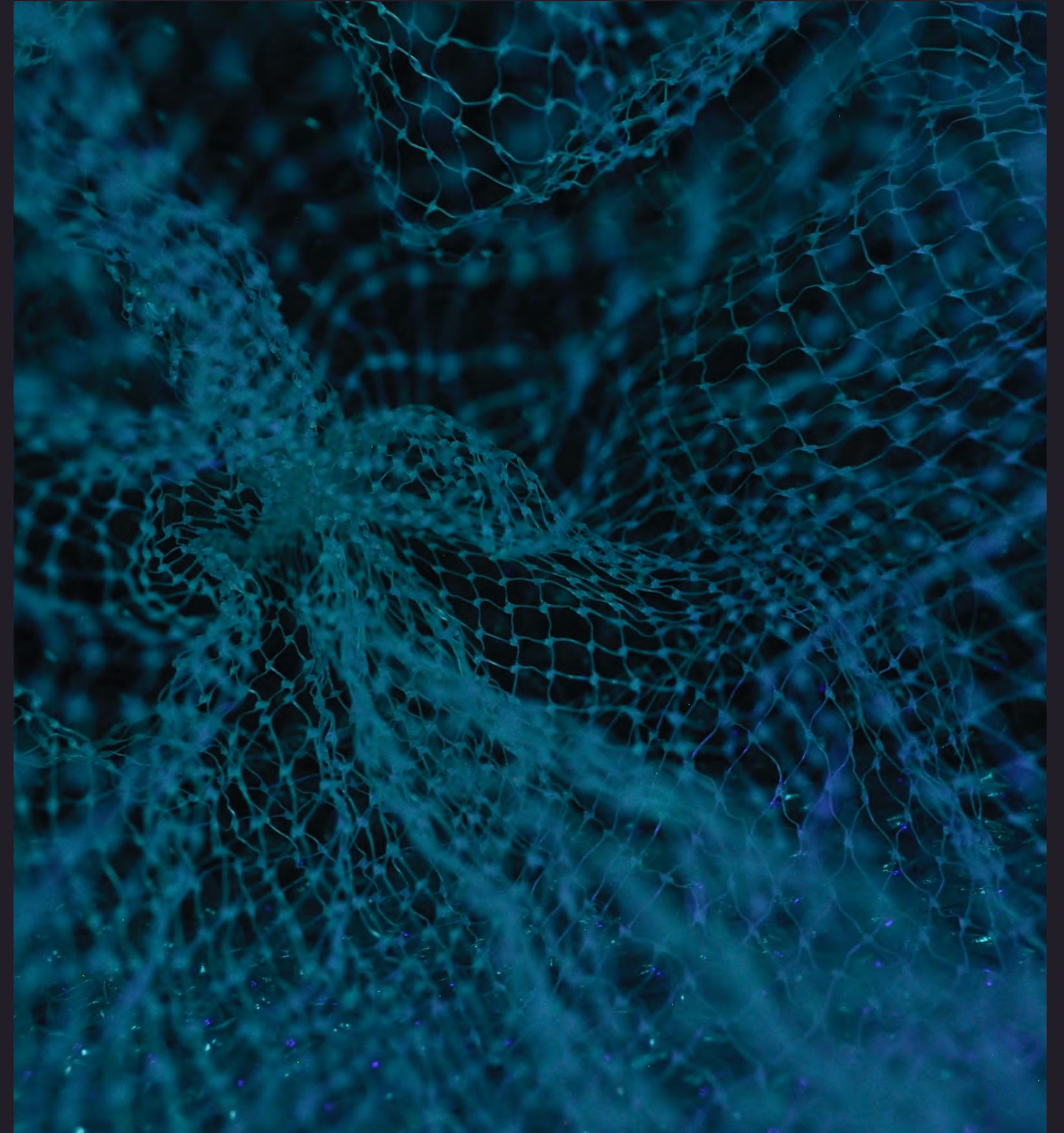
Advantages/ features

- **Decentralized Hosting:** Bypass centralized entities; host independently, reducing vulnerabilities and centralised points of failure.
- **Transparent Interactions:** Ensure clear, trust-based interactions between the community and service provider.
- **Universal Browser Compatibility:** Access the blockchain-hosted websites from standard browsers, ensuring easy accessibility .
- **Unified Domain Approach:** Use a primary domain (the node's/chain's main domain) for hosting, co located by tx hash or a name service.
- **Global Content Distribution:** Multiple nodes host identical website content, promoting content decentralisation.
- **Swift Access:** Much faster access, files are just being read like in block explorer



Future Improvements

- **Improved CLI:** fix broken cli, and making it user-friendly even for those with minimal technical expertise.
- **Website Upgradability:** Implement a feature that allows easy updates and modifications to websites without starting from scratch.
- **Name Service Integration:** Allow users to access sites using human-readable names instead of complex transaction hashes.
- **Support for IPFS & File-coin:** Integrate decentralised storage solutions like IPFS and FileCoin to enhance storage capabilities and redundancy.
- **JS SDK:** Develop a JavaScript Software Development Kit to simplify and streamline the website deployment process for developers familiar with JS.
- **Others:** support for Security Certificates. support token transfers as utxo, native support from blockchain for Creation of tokens instead of smart contract



Demo (Some of the parts of app are incomplete)

- 1. Build And Compile Your project.

A simple staking dapp will be used with contracts deployed on polygon testnet
<https://github.com/apoorv-2204/staking-dapp>

cd client and run npm run build

- 2. Ensure that build has index.html at root and able to be served by simple live server extension.

Copy the build folder and do some tweaks util it can be hosted via live server.

```
apoorv-2204@apoorv-laptop:~/Documents/100builder/dapp$ tree -L 2
.
├── index.html
├── _next
│   ├── BUILD_ID
│   ├── build-manifest.json
│   ├── cache
│   ├── export-marker.json
│   ├── images-manifest.json
│   ├── next-server.js.nft.json
│   ├── package.json
│   ├── prerender-manifest.json
│   ├── react-loadable-manifest.json
│   ├── required-server-files.json
│   ├── routes-manifest.json
│   ├── server
│   ├── static
│   └── trace
└──
```

4 directories, 12 files

STAKING-DAPP

> artifacts

> cache

> client

> .next

> components

> constants

ContractInfo.json

RewardToken.json

Staking.json

> node_modules

> pages

> public

> styles

.babelrc

.eslintrc.json

.gitignore

next.config.js

package-lock.json

package.json

postcss.config.js

tailwind.config.js

yarn.lock

> contracts

> node_modules

> scripts

> test

.env

.gitignore

hardhat.config.js


notes.md

package-lock.json

package.json

README.md

OUTLINE



Live Server

v5.7.1

Ritwick Dey

38,904,081

Launch a development local

Disable

Uninstall

⚙

This extension is enabled global

Deploy

3. Go to <https://github.com/apoorv-2204/Native-DWeb-Hosting> , clone switch to master.
run cmd ignite chain serve
4. Run the cli cmd Native-Dweb-Hostingd
currently broken but it should allow to deploy a tx
with website after that you will get a tx hash

ex:

FCA1AFE3E8729EF44E66E092464EC0F0FEAAE392568F
167144D462B95156BD5A

5. Request the node endpoint
localhost:1317/hosting/FCA1AFE3E8729EF44E66E092
464EC0F0FEAAE392568F167144D462B95156BD5A
6. Currently More work is required to improve it.Currently
manually is done until cli is complete, as most of the
parts are not complete

```
apoorv-2204@apoorv-laptop: ~/Documents/100builder/Native-DWeb-Hosting/cmd/Native-DWeb-Ho
Start natedwebhosting node

Usage:
  Native-DWeb-Hostingd [command]

Available Commands:
  add-genesis-account Add a genesis account to genesis.json
  collect-gentxs       Collect genesis txs and output a genesis.json file
  config              Create or query an application CLI configuration file
  debug               Tool for helping with debugging your application
  export              Export state to JSON
  gentx               Generate a genesis tx carrying a self delegation
```

