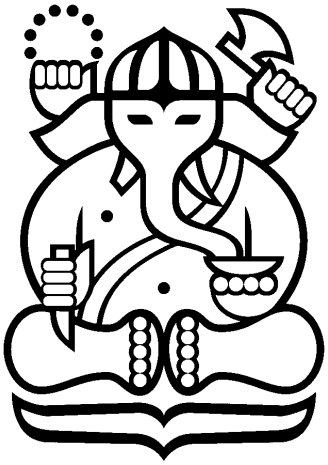
PROJECT REPORT

IF4035 BLOCKCHAIN



Dibuat Oleh:

Christian Albert Hasiholan / 13521078
Tobias Natalio Sianipar / 13521090
Haidar Hamda / 13521105

PROGRAM STUDI TEKNIK INFORMATIKA
SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA
INSTITUT TEKNOLOGI BANDUNG
BANDUNG

1. Problem Statement

Pembayaran royalti seringkali menjadi hambatan bagi para pembuat konten, baik dari keterlambatan pembayaran atau pembayaran yang tidak sesuai nilainya. Untuk itu, pada tugas ini diimplementasikan blockchain untuk melakukan pembayaran royalti sesuai dengan jumlah penggunaan suatu konten (*views*) dan *smart contract* yang telah ditetapkan.

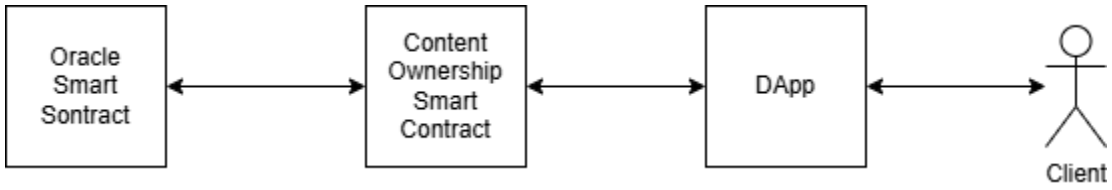
2. Platform

Platform yang dipilih dalam implementasi private blockchain ini merupakan Geth (Private Ethereum), lebih tepatnya dengan versi 1.13.15. Geth dipilih sebagai platform blockchain untuk mencatat dan membayar royalti karena kemampuan mencatat transaksi secara permanen di blockchain, Geth memastikan semua pembayaran royalti tercatat secara transparan dan tidak dapat diubah, sehingga memberikan akuntabilitas tinggi kepada semua pihak yang terlibat. Dengan penggunaan smart contract juga, pembayaran royalti dapat diotomatisasi berdasarkan aturan yang telah ditentukan, seperti pembagian royalti berdasarkan jumlah penggunaan konten (*views*), sehingga mengurangi ketergantungan pada proses manual. Selain itu, Geth memungkinkan pengelolaan royalti di private blockchain, memberikan kontrol atas kerahasiaan data serta efisiensi dalam mekanisme konsensus menggunakan Clique yang berdasarkan Proof of Authority (PoA). Proses pembayaran juga menjadi lebih cepat karena Ether dapat ditransfer langsung ke penerima royalti tanpa perantara, sementara seluruh transaksi tetap dapat diaudit secara transparan.

3. Tech Stack

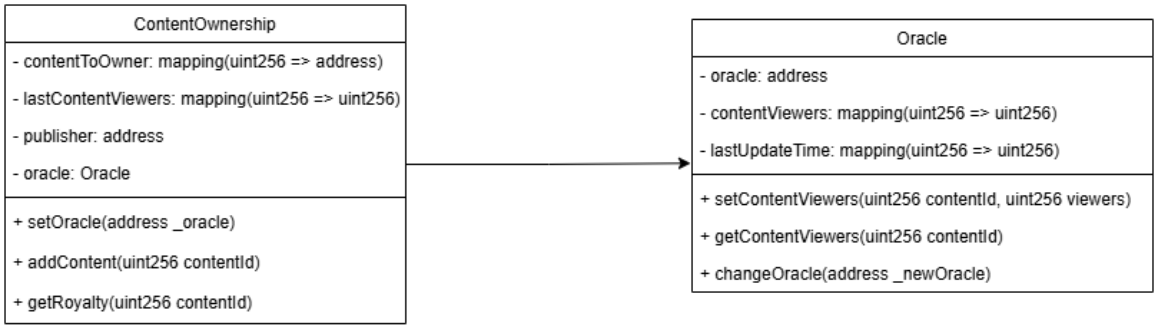
- Geth: Platform implementasi private blockchain berdasarkan Ethereum
- Truffle: Framework pengembangan blockchain untuk membangun, menguji, dan deploy smart contract di Ethereum.
- Solidity: Bahasa pemrograman yang digunakan untuk menulis smart contract di Ethereum.
- Metamask: Cryptocurrency wallet berbasis browser extension yang memungkinkan pengguna untuk berinteraksi dengan dApps di blockchain.
- Next: framework yang digunakan untuk frontend
- Web3: API ethereum. Digunakan untuk akses smart contract yang ada.

4. High Level Desain



Client berinteraksi dengan DApp. Dalam DApp, client dapat melakukan penambahan content dan penarikan pendapatan berdasarkan jumlah view dari content yang ditambahkan. Ketika DApp mendapatkan perintah dari client untuk melakukan penambahan atau penarikan, DApp mengeksekusi fungsi yang sesuai pada smart contract menggunakan API (web3). Setiap node melakukan validasi untuk setiap transaksi yang ada. Jika terjadi perintah penarikan, content ownership smart contract akan mengambil data jumlah view dari oracle.

5. Properti dan Metode Smart Contract



Terdapat 2 Smart Contract yang diimplementasikan dalam blockchain ini, yaitu ContentOwnership dan Oracle. ContentOwnership digunakan oleh para pemilik konten untuk menambahkan kontennya dan memperoleh royalti dari konten mereka dengan informasi *view* melalui Oracle. Sedangkan Oracle digunakan untuk memperoleh jumlah *view* konten pada sebuah platform, misalnya pada Youtube atau Spotify, namun disini jumlah view suatu konten diset secara manual untuk mensimulasikan kerja Oracle.

6. Oracle

a. Strukur Data

Oracle mengambil jumlah view pada suatu konten dengan id tertentu dan menyimpannya sebagai *map* {content id, views} pada *smart contract*.

b. Mekanisme Pengambilan Data

Data disimpan pada *smart contract* dengan menggunakan API *truffle* untuk mengeksekusi fungsi *pada smart contract* yang hanya bisa dieksekusi bila address yang

digunakan adalah address oracle yang di inisiasi pada saat deploy ke *blockchain*. Fungsi tersebut menginput content id dan jumlah view sekarang terhadap konten tersebut.

c. Skenario Buruk

Melakukan iterasi melalui array atau mapping besar yang disimpan di dalam kontrak akan mengakibatkan *gas fee* untuk setiap operasi *write* atau *read*. Struktur yang lebih besar berarti lebih banyak operasi, dan akibatnya biaya yang lebih tinggi. Oleh karena itu diimplementasi suatu *timestamp guard* yang dapat me-*limit* frekuensi *write* agar data yang disimpan tidak terlalu banyak.

7. Design Pattern

Design pattern yang digunakan adalah *access control* dimana hanya *account* yang memenuhi *onlyOracle* yang dapat menulis jumlah *views* suatu konten pada *smart contract* oracle sedangkan *account* lain hanya dapat mengakses *getContentViewsI*. Sama halnya dengan *smart contract* content provider yang memerlukan *account content owner* dari suatu konten untuk mendapatkan uang dari konten tersebut.

Design pattern access control dipilih untuk meningkatkan flexibilitas dari *smart contract* yang diimplementasi sehingga berbagai macam *account* dapat menggunakan *smart contract* tersebut dengan akses yang terkontrol.

Daftar anggota kelompok

NIM	Nama	Tugas
13521078	Christian Albert Hasiholan	Private Network, Smart Contract, Oracle, Wallet
13521090	Tobias Natalio Sianipar	Private Network, Smart Contract, Oracle, Wallet
13521105	Haidar Hamda	Private Network, Smart Contract, Oracle, Wallet, FE

Referensi

https://geth.ethereum.org/docs/fundamentals/private-network
https://ferdyhape.medium.com/blockchain-for-beginners-build-your-own-private-ethereum-network-with-geth-step-by-step-311342370fec
https://medium.com/coinmonks/dapp-on-a-private-ethereum-network-1-c8b80695e049

Lampiran

Repo Github: https://github.com/haidarhamda/tubes-blockchain
Video Demo: Demo Tubes Blockchain.mp4