Implementasi Teknologi Blockchain pada *Real Estate Transaction*

Ceavin Rufus De Prayer Purba (18221162)
Program Studi Sistem dan Teknologi Informasi
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
E-mail (gmail): ceavinrdpp@gmail.com

***Abstract*—Teknologi *blockchain* menawarkan solusi inovatif untuk mengatasi berbagai permasalahan dalam industri *real estate*. Blockchain memberikan transparansi, keamanan, dan pencatatan transaksi *real estate* yang aman dan permanen. Penggunaan *smart contracts*, NFT, dan IPFS membantu otomatisasi, representasi kepemilikan, dan penyimpanan metadata properti. Penerapan *blockchain* dalam transaksi *real estate* memiliki banyak manfaat yang meliputi peningkatan transparansi, pengurangan risiko penipuan, percepatan proses transaksi, efisiensi, dan mendukung inovasi. Tantangan yang dihadapi termasuk pemahaman teknologi yang kurang, regulasi, dan infrastruktur. Meski demikian, penerapan blockchain berpotensi merevolusi industri real estate.**

Keywords—blockchain; real estate; smart contracts; transparansi; keamanan; escrow contracts, NFT

I. Pendahuluan

Pada tahun 2020, nilai aset *real estate* naik 5% secara global hingga mencapai $326,5 triliun. Hal ini menjadikan *real estate* sebagai sumber kekayaan paling signifikan di dunia, melampaui semua ekuitas dan surat utang global yang digabungkan, serta hampir empat kali lipat dari Produk Domestik Bruto (PDB) global [1]. *Real estate* adalah sektor yang melibatkan transaksi jual beli properti nyata, mengacu pada tanah beserta segala sesuatu yang berada di atas atau di dalamnya, seperti bangunan dan sumber daya alam terkait [2].

Namun, transaksi *real estate* yang melibatkan banyak pihak seperti agen properti, notaris, bank, dan lembaga pemerintah membuat transaksi menjadi kompleks dan memiliki potensi untuk menghambat proses secara keseluruhan. Transparansi dalam pencatatan transaksi dan kepemilikan properti yang kurang transparan pun meningkatkan risiko terjadinya manipulasi data serta penipuan. Selain itu, informasi transaksi menjadi rentan terhadap serangan siber dan peretasan dikarenakan informasi transaksi disimpan dalam sistem yang terpusat. Dalam memasuki pasar *real estate* juga membutuhkan modal yang besar, sehingga investor kecil pun sulit mendapatkan akses.

Implementasi teknologi *blockchain* dalam transaksi *real estate* dapat menawarkan solusi potensial untuk mengatasi berbagai masalah pada industri *real estate*. *Blockchain* adalah teknologi buku besar yang terdistribusi dan dapat memfasilitasi proses pencatatan transaksi dan pelacakan aset dalam jaringan bisnis [3]. Sistem ini bekerja dengan menciptakan catatan yang didistribusikan di berbagai *server*, sehingga manipulasi dan serangan siber yang sering terjadi pada sistem terpusat dapat diminimalkan.

Penerapan teknologi *blockchain* dalam transaksi *real estate* memiliki banyak manfaat, salah satunya adalah peningkatan transparansi dan keamanan. Setiap transaksi yang dicatat dalam *blockchain* akan dapat diakses oleh semua pihak yang terlibat dan data bersifat permanen, sehingga dapat meminimalkan terjadinya permasalahan mengenai manipulasi data maupun penipuan. Tak hanya itu, *smart contracts* atau kontrak digital yang secara otomatis dieksekusi ketika kondisi terpenuhi juga dapat diterapkan dengan penggunaan teknologi *blockchain*. Hal ini dapat mengurangi kebutuhan akan perantara seperti agen properti dan notaris serta mempercepat proses transaksi yang berdampak dalam menurunkan biaya transaksi *real estate*.

Blockchain dapat mengubah cara dalam penjualan, pembelian serta pengelolaan properti dengan mengatasi tantangan yang ada dan meningkatkan efisiensi serta keamanan. Dalam jangka panjang, penerapan teknologi *blockchain* memiliki potensi untuk mendorong pertumbuhan ekonomi dan inovasi lebih lanjut di sektor industri *real estate*. Maka dari itu, makalah ini akan membahas terkait implementasi teknologi *blockchain* pada sebuah platform transaksi *real estate*.

II. Tinjauan Pustaka

A. Blockchain

Blockchain adalah buku besar terdistribusi yang dikelola secara kolektif dan tidak terpusat sehingga tidak ada *server* tunggal yang memiliki akses penuh ke seluruh data. Sebaliknya, data pada *blockchain* tersebar di banyak *server* [4]. Basis data *blockchain* menyimpan data dalam blok yang terhubung dalam rantai, sehingga data tidak dapat dihapus atau diubah tanpa persetujuan jaringan [5].

Dalam jaringan *blockchain*, setiap transaksi dicatat dalam sebuah blok, yang kemudian dirantai dengan blok lainnya dalam urutan kronologis menggunakan algoritma kriptografi yang kuat untuk memastikan bahwa setiap blok valid dan tidak dapat diubah. Semua *node* di jaringan harus mencapai konsensus untuk menambahkan blok baru ke dalam rantai setiap kali ada transaksi baru. Hal ini membuat *blockchain* sangat aman dan sulit untuk dimanipulasi, karena seseorang

harus memiliki kontrol atas sebagian besar jaringan, yang biasanya sangat tidak mungkin dilakukan.

Transparansi adalah salah satu manfaat utama *blockchain* selain keamanannya. Untuk mengurangi risiko penipuan dan meningkatkan kepercayaan jaringan, semua transaksi yang tercatat dalam *blockchain* dapat dilihat oleh siapa saja yang memiliki akses ke jaringan tersebut. Berbagai aplikasi, seperti keuangan, contohnya mata uang kripto ataupun logistik, pengelolaan identitas, dan *smart contract*, dapat menggunakan teknologi ini.

B. *Real Estate*

Real estate adalah segala properti nyata, seperti tanah dan segala struktur permanen yang berada di atasnya, baik yang alami maupun buatan manusia [6]. Industri *real estate* berperan penting dalam investasi, pengembangan wilayah, serta penyediaan hunian dan komersial sehingga memiliki dampak yang signifikan terhadap perekonomian global.

Berdasarkan penelitian mengenai dampak dari harga properti, pasar *real estate* mempengaruhi stabilitas finansial dan sosial masyarakat [7]. Industri *real estate* pun telah mengalami perubahan besar seiring berjalannya perkembangan teknologi, termasuk dalam aspek manajemen properti dan pemasaran [8]. *Real estate* adalah sektor yang kompleks dan dinamis dengan banyak faktor yang mempengaruhi pertumbuhannya dari waktu ke waktu.

C. Ethereum

Ethereum adalah jaringan komputer terbuka yang terdesentralisasi dan menggunakan teknologi *blockchain* serta berfungsi sebagai dasar untuk komunitas, organisasi, aplikasi, maupun aset digital yang dapat dikembangkan dan dimanfaatkan oleh semua orang [9]. Ethereum tidak dikelola oleh satu entitas tunggal, melainkan oleh jaringan komputer global (*node*) yang bekerja sama untuk memvalidasi dan mengamankan transaksi.

Dengan menggunakan Ethereum, maka pengembang dapat membuat berbagai jenis aplikasi melalui *smart contracts* yang terdapat pada *blockchain* Ethereum. Selain itu, Ethereum juga mendukung dan memfasilitasi pembuatan dan perdagangan *Non Fungible Token* (NFT) yang merupakan aset digital di dunia nyata yang dapat mewakili suatu objek tertentu [10].

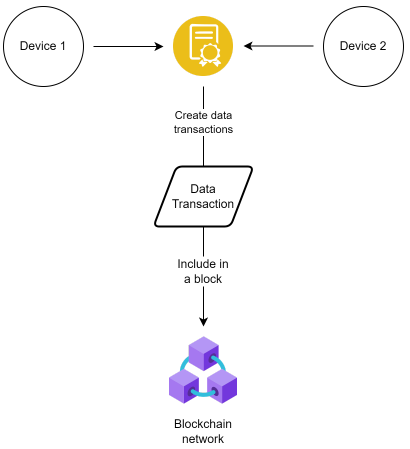
D. *Non Fungible Token*

Non Fungible Token (NFT) adalah aset digital yang dapat berbentuk seni ataupun objek pada sebuah permainan dengan basis *blockchain* Ethereum[11]. NFT diperjualbelikan secara *online*, sering kali menggunakan *cryptocurrency*. NFT tidak hanya dapat terhubung dengan seni, tetapi juga dengan teks, video, atau potongan kode.

NFT semakin diminati karena memberikan peluang kepemilikan digital yang eksklusif dan sah atas beragam jenis aset. Salah satu kelebihan utama NFT adalah mampu memberikan kepastian akan kepemilikan dan keaslian suatu aset digital. Selain itu, NFT juga memberikan peluang kepada pencipta konten digital, seperti seniman atau pengembang permainan, untuk meraih penghargaan dan imbalan atas karya yang mereka hasilkan.

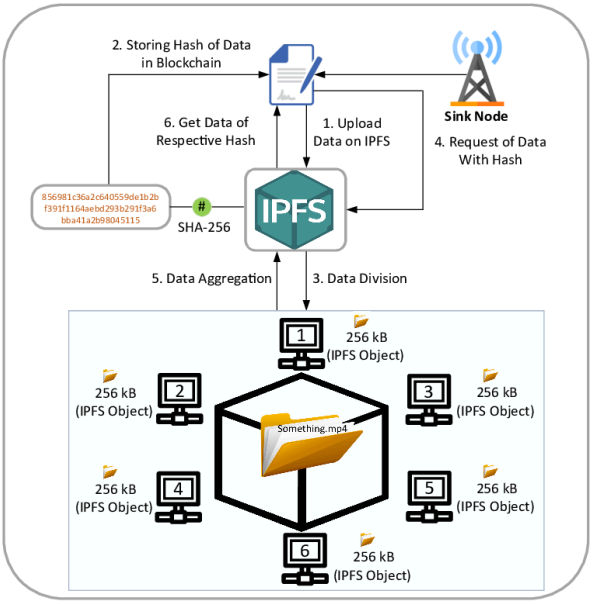
E. *Smart Contract*

Smart contract perjanjian atau kontrak digital yang tersimpan di dalam blockchain dan secara otomatis mengeksekusi, menegosiasikan, atau menegakkan kesepakatan ketika kondisi yang ditentukan telah tercapai [12]. Dengan menggunakan teknologi *blockchain*, smart contract dapat berjalan tanpa adanya pihak ketiga dan secara desentralisasi serta tidak dapat diubah atau dimanipulasi. Kemampuan ini menjadikan *smart contract* sebagai alat yang efektif untuk mengotomatisasi dan mengamankan berbagai jenis transaksi.



Salah satu keunggulan dari *smart contract* adalah tidak memungkinkan terjadinya kesalahan manusia atau interpretasi yang salah dikarenakan *smart contract* bekerja sesuai dengan kode yang ditulis. *Smart contract* juga bersifat transparan dikarenakan dapat dipantau oleh semua pihak yang terlibat. Terakhir, dengan mengotomatisasi proses, penggunaan *smart contract* dapat menghemat waktu serta sumber daya yang digunakan.

F. *InterPlanetary File System (IPFS)*



InterPlanetary File System (IPFS) adalah kumpulan protokol yang dapat disesuaikan untuk mengatur dan

mengirim data, dibangun dengan prinsip-prinsip pengalamatan konten dan jaringan *peer-to-peer*. Meskipun IPFS memiliki beberapa aplikasi dalam penggunaannya, tujuan utama IPFS adalah untuk memungkinkan publikasi data secara terdesentralisasi, termasuk *file*, direktori, dan situs web [13].

IPFS memanfaatkan teknologi *blockchain* untuk mencatat dan memverifikasi keaslian konten. Setiap *file* diberikan *hash* unik yang memungkinkan identifikasi dan akses salinan *file* dari berbagai sumber di seluruh jaringan. Berbeda dengan model penyimpanan *server* tradisional yang mengandalkan *server* pusat untuk menyimpan konten, IPFS mendistribusikan konten secara luas di seluruh jaringan, menjadikannya lebih tangguh terhadap kegagalan dan upaya sensor.

G. ERC-721 *Non Fungible Token Standard*

ERC-721 adalah standar *smart contract* pada *blockchain* Ethereum yang digunakan untuk menciptakan dan mengelola token *non-fungible* (NFT) [14]. ERC-721 menentukan fungsi dan aturan yang harus dimiliki oleh smart contract untuk menciptakan NFT. Seperti kemampuan untuk mentransfer token dari satu akun ke akun lain, memeriksa saldo token terkini pada suatu akun, menemukan pemilik dari token tertentu, dan juga mengetahui total pasokan token yang terdapat di jaringan. Selain itu, ERC-721 juga dilengkapi dengan beberapa fungsi lain seperti memberikan persetujuan bahwa sejumlah token dari suatu akun dapat dipindahkan oleh akun lain (pihak ketiga).

H. *Escrow Contract*

Escrow contract adalah kesepakatan kontrak antara dua atau lebih pihak sebagai perantara dalam sebuah transaksi [15]. Tujuan utama *escrow contract* adalah untuk menjamin bahwa transaksi antara dua pihak atau lebih berlangsung dengan aman dan efisien. *Escrow contract* mengambil alih kontrol atas aset yang terlibat dalam transaksi dan memberikannya kepada pihak yang sesuai dengan persyaratan yang telah ditetapkan.

Escrow contract sering digunakan dalam berbagai transaksi seperti pembelian *online* atau properti. Dalam aspek *blockchain*, *escrow contract* diimplementasikan dengan *smart contract*, seperti yang terdapat pada *blockchain* Ethereum, untuk meningkatkan tingkat keamanan dan transparansi dalam pengelolaan transaksi. Dengan mengurangi risiko penipuan atau perselisihan antara pihak yang terkait, *escrow contract* membantu mendorong pertumbuhan bisnis *online* dan perdagangan *peer-to-peer* yang aman dan dapat diandalkan.

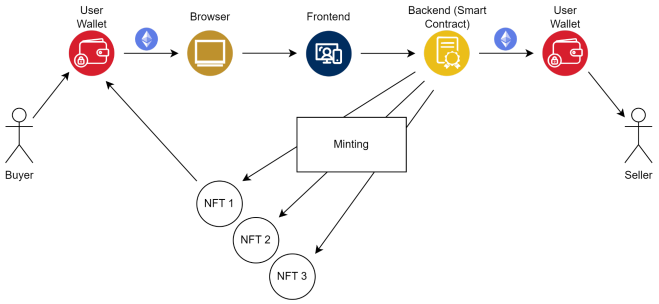
I. *User Wallet*

User wallet atau bisa juga disebut sebagai dompet kripto adalah alat atau aplikasi yang digunakan oleh pengguna untuk menyimpan, mengirim, dan menerima mata uang kripto seperti Bitcoin, Ethereum, atau koin kripto lainnya [16]. *User wallet* memfasilitasi akses pengguna ke saldo mereka dalam bentuk *cryptocurrency* dan memungkinkan mereka untuk melakukan transaksi dengan cepat dan mudah.

User wallet bisa dalam bentuk aplikasi yang dipasang di perangkat seluler atau komputer, maupun layanan *online* yang diakses melalui *web browser*. Beberapa dari *user wallet* dioperasikan secara terdesentralisasi, yang berarti pengguna memiliki kendali penuh atas kunci pribadi mereka, sedangkan yang lain bisa dioperasikan oleh perusahaan yang menyimpan kunci pribadi pengguna atas nama mereka.

III. Desain dan Rancangan

A. *Skema Transaksi Secara Umum*



Proses transaksi *real estate* menggunakan teknologi *blockchain* dimulai dengan setiap pihak yang terlibat, yaitu pembeli dan penjual, memiliki dompet digital (*wallet*) yang menyimpan *cryptocurrency*, dalam hal ini menggunakan Ethereum (ETH). *Wallet* ini digunakan untuk mengelola aset kripto dan memastikan otentikasi serta verifikasi identitas dalam setiap transaksi. Platform transaksi *real estate* yang berjalan di atas server mengelola data dan informasi terkait properti yang akan diperjualbelikan, termasuk detail pemilik, harga, lokasi, dan legalitas properti. Pengguna dalam platform ini memiliki identitas digital yang telah diverifikasi melalui proses KYC (Know Your Customer) dan sertifikat digital yang mengonfirmasi keabsahan serta kepemilikan aset.

Proses *minting* dari NFT (Non-Fungible Token) dimulai setelah verifikasi identitas dan legalitas properti selesai. NFT ini mencerminkan kepemilikan properti yang unik dan menyimpan informasi lengkap mengenai properti tersebut, termasuk pemilik, harga, dan sejarah transaksi. Dengan menggunakan teknologi *blockchain*, NFT akan dibuat secara unik untuk setiap properti. Selain itu, data yang dimasukkan ke dalam NFT dienkripsi untuk memastikan keamanan dan integritasnya. *Blockchain* memastikan bahwa data ini tidak dapat diubah atau dihapus setelah dicetak, sehingga NFT tidak dapat dipalsukan. Hal ini memberikan kepastian hukum dan kepercayaan dalam transaksi *real estate*. Proses ini melibatkan pembuatan token digital yang mewakili properti tersebut di *blockchain*. Setelah NFT dicetak, transaksi jual beli properti dapat dilakukan. Pembeli mengirimkan *cryptocurrency* (ETH)

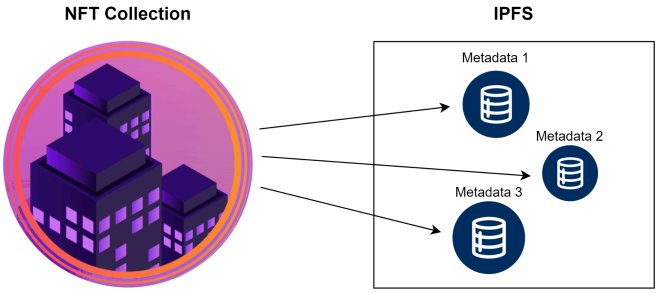
kepada penjual, dan sebagai gantinya, kepemilikan NFT yang mewakili properti tersebut ditransfer ke pembeli.

Untuk mengatur transaksi ini, digunakan sebuah backend berupa *smart contract*. *Smart contract* digunakan untuk mengelola dan mengeksekusi transaksi. *Smart contract* menyimpan aturan dan logika transaksi, seperti verifikasi pembayaran dan transfer kepemilikan properti, yang akan dieksekusi secara otomatis ketika kondisi tertentu terpenuhi. Misalnya, ketika seorang pembeli menyelesaikan pembayaran, *smart contract* akan otomatis mentransfer kepemilikan NFT dari penjual ke pembeli, sekaligus memperbarui catatan kepemilikan di *blockchain*.

Selain untuk menyimpan *cryptocurrency*, *wallet* yang dimiliki user juga berfungsi sebagai alat untuk menyimpan aset digital lainnya, seperti *tokenized real estate*, dengan aman. Setelah transaksi selesai, NFT yang mewakili kepemilikan properti sekarang ada di wallet pembeli, dan penjual menerima pembayaran dalam bentuk *cryptocurrency*. Semua detail transaksi, termasuk transfer kepemilikan dan pembayaran, dicatat secara permanen di *blockchain*. Teknologi *blockchain* memungkinkan untuk membuat catatan transparan dan tidak dapat diubah, serta dapat diverifikasi oleh semua pihak yang berkepentingan.

Dengan cara ini, seluruh proses transaksi properti menjadi lebih efisien, aman, dan transparan, mengurangi kebutuhan akan perantara tradisional dan meminimalkan risiko pemalsuan atau sengketa kepemilikan.

B. Penyimpanan Metadata pada IPFS



Proses penyimpanan data pada NFT menggunakan IPFS (*InterPlanetary File System*) dimulai dengan pengumpulan metadata dari properti *real estate*, seperti gambar, deskripsi, informasi pemilik, dan detail lainnya. Data ini disusun dalam format JSON atau format metadata yang didukung oleh standar NFT, seperti ERC-721 yang akan digunakan pada implementasi yang akan dilakukan, atau standar lain seperti ERC-1155.

Setiap metadata mewakili informasi unik mengenai properti yang diwakili oleh NFT. Misalnya, Metadata 1 mungkin berisi informasi tentang sebuah apartemen, Metadata 2 tentang sebuah rumah, dan Metadata 3 tentang sebidang tanah.

Setelah metadata dikumpulkan, data tersebut diunggah ke jaringan IPFS. Di sini, data tersebut dibagi menjadi blok-blok yang terdistribusi di berbagai *node* dalam jaringan IPFS. Setiap *file* atau blok data yang diunggah ke IPFS diberikan *content identifier* (CID) yang unik. CID berfungsi sebagai alamat permanen dari data tersebut.

CID memastikan bahwa data dapat diakses dan diverifikasi tanpa bergantung pada lokasi penyimpanan tertentu. Hal ini berarti bahwa meskipun lokasi penyimpanan fisik dari data mungkin berubah, CID akan tetap sama, memungkinkan pengguna untuk dengan mudah mengakses dan memverifikasi data NFT yang terkait dengan properti *real estate*.

Dalam konteks standar ERC-721, setiap NFT mewakili satu properti yang unik. Artinya, jika kita memiliki tiga properti *real estate* yang berbeda, kita memiliki tiga NFT.

Berikut adalah contoh metadata terkait properti yang disimpan pada IPFS dalam JSON format:

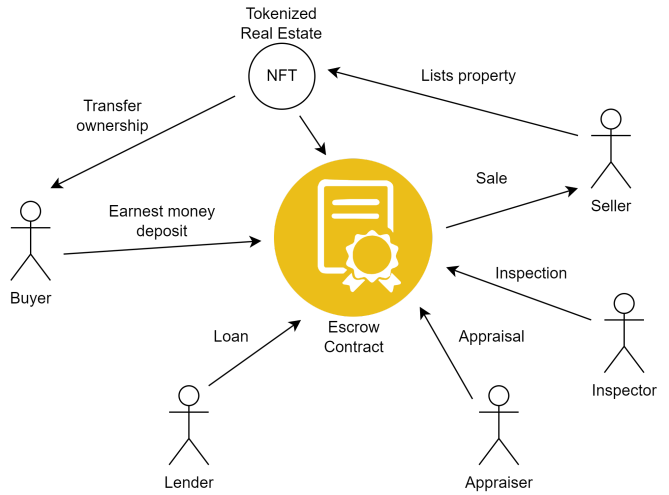
```
{
  "name": "Luxurious Apartment in Jakarta City
Center",
  "description": "A stunning apartment located in
the heart of Jakarta, offering breathtaking views
of the city skyline.",
  "image":
"https://example.com/apartment_image.jpg",
  "external_url":
"https://example.com/apartment123",
  "attributes": [
    {
      "trait_type": "Type",
      "value": "Apartment"
    },
    {
      "trait_type": "Size (sqm)",
      "value": 150
    },
    {
      "trait_type": "Bedrooms",
      "value": 3
    },
    {
      "trait_type": "Bathrooms",
      "value": 2
    },
    {
      "trait_type": "Floor",
      "value": 20
    }
  ],
  "properties": {
    "city": "Jakarta",
    "country": "Indonesia",
    "owner_name": "John Doe",
    "owner_email": "johndoe@example.com",
    "ethereum_address":
"0x123456789abcdef0123456789abcdef01234567",
    "token_name": "LuxuryApartmentToken",
    "token_symbol": "LAT",
    "total_supply": 1000,
    "contract_address":
"0x987654321fedcba0987654321fedcba09876543",
    "token_id": 123
  }
}
```

Setelah metadata diunggah ke IPFS dan mendapatkan CID, CID ini disertakan dalam NFT *contract*, yang menghubungkan NFT dengan metadata yang tersimpan di IPFS. Ini memungkinkan setiap NFT yang dicetak mengandung tautan ke metadata yang dapat diakses kapan saja oleh pemiliknya. Saat NFT diperdagangkan, pemilik baru dapat mengakses metadata menggunakan CID yang disertakan dalam NFT, dengan jaminan bahwa data tersebut adalah versi asli dan tidak berubah sejak diunggah.

Penggunaan IPFS memberikan beberapa keuntungan signifikan, yaitu:

- Desentralisasi, yang memastikan data disimpan dalam jaringan terdistribusi sehingga lebih tahan terhadap kegagalan sentral dan sensor;
- Keamanan dan integritas data yang dijamin oleh CID unik yang memastikan data tidak dapat diubah tanpa mengubah CID-nya;
- Penyimpanan permanen, yang memungkinkan data terkait NFT dapat diakses secara aman dan permanen.

C. Penggunaan Escrow Contract



Dalam implementasi *blockchain* pada transaksi *real estate* seperti yang digambarkan di atas, penggunaan *escrow contract* menjadi krusial dalam mengamankan dan memfasilitasi proses tersebut. *Escrow contract* pada *blockchain* bertindak sebagai entitas netral yang mengelola dana transaksi di dalam *smart contract*. Ketika pembeli memberikan deposito, transaksi tersebut tercatat dalam *escrow contract* yang dijalankan di atas platform *blockchain*. Hal ini memastikan bahwa dana tersebut aman dan tidak akan dilepaskan sampai semua persyaratan *contract* terpenuhi.

Selain itu, *escrow contract* pada *blockchain* juga mengotomatiskan sebagian besar proses transaksi. Misalnya, begitu *appraiser* memberikan penilaian nilai properti atau inspektur menyelesaikan pemeriksaan fisik, informasi tersebut dapat secara langsung diunggah ke dalam *escrow contract*. Hal ini meminimalkan risiko manipulasi data dan memastikan transparansi dalam proses tersebut.

Selain memfasilitasi transaksi, *escrow contract* pada *blockchain* juga membantu dalam menegakkan kepatuhan terhadap peraturan dan persyaratan *contract*. Semua pihak terlibat, termasuk penjual, pembeli, dan pemberi pinjaman (*lender*), dapat melihat dan memverifikasi setiap langkah dalam proses transaksi, sehingga mengurangi potensi konflik atau sengketa di masa mendatang.

Jadi, secara singkat berikut adalah alur proses pada pelaksanaan transaksi dengan menggunakan *escrow contract*:

1. Penjual memasukkan daftar properti yang akan dijual.
2. Pembeli menyetor uang jaminan (*earnest money deposit*) ke *escrow contract* sebagai tanda keseriusan.
3. Penilai (*appraisal*) memeriksa properti dan memberikan penilaian nilainya.
4. *Inspector* melakukan pemeriksaan menyeluruh terhadap kondisi properti.
5. Pemberi pinjaman (*lender*) menilai dan menyetujui pinjaman yang diminta pembeli.
6. Pemberi pinjaman menyediakan dana pinjaman yang diperlukan.
7. Setelah semua persyaratan terpenuhi, kepemilikan properti ditransfer dari penjual ke pembeli.
8. Penjual menerima hasil penjualan dari *escrow contract*.

Adapun berikut adalah contoh hasil catatan transaksi oleh *escrow contract* dalam format JSON:

```
[
    {
        "inputs": [
            {
                "internalType": "address",
                "name": "_nftAddress",
                "type": "address"
            },
            {
                "internalType": "address payable",
                "name": "_seller",
                "type": "address"
            },
            {
                "internalType": "address",
                "name": "_inspector",
                "type": "address"
            },
            {
                "internalType": "address",
                "name": "_lender",
                "type": "address"
            }
        ],
        "stateMutability": "nonpayable",
        "type": "constructor"
    }
]
```

Dengan menggunakan *escrow contract* pada proses transaksi *real estate*, proses transaksi akan lebih efisien dan

transparan. Selain itu, penggunaan *escrow contract* juga akan mengurangi risiko serta meningkatkan kepercayaan di antara semua pihak yang terlibat.

IV. Implementasi

A. Implementasi RealEstate Smart Contract

```
//SPDX-License-Identifier: Unlicense
pragma solidity ^0.8.0;

import "@openzeppelin/contracts/utils/Counters.sol";
import "@openzeppelin/contracts/token/ERC721/ERC721.sol";
import "@openzeppelin/contracts/token/ERC721/extensions/ERC721URIStorage.sol";

contract RealEstate is ERC721URIStorage {
    using Counters for Counters.Counter;
    Counters.Counter private _tokenIds;

    constructor() ERC721("Real Estate", "REAL") {}

    function mint(string memory tokenURI) public returns (uint256) {
        _tokenIds.increment();

        uint256 newItemId = _tokenIds.current();
        _mint(msg.sender, newItemId);
        _setTokenURI(newItemId, tokenURI);

        return newItemId;
    }

    function totalSupply() public view returns (uint256) {
        return _tokenIds.current();
    }
}
```

Secara garis besar, *smart contract* ini memiliki fungsi utama untuk menciptakan dan mengelola *real estate* sebagai NFT di Ethereum. *Smart contract* ini memungkinkan pengguna untuk menciptakan token *real estate* yang unik sebagai NFT di Ethereum. Setiap token mewakili kepemilikan atas properti *real estate* tertentu dan dapat ditautkan ke URI untuk menyimpan informasi tambahan tentang properti tersebut, seperti gambar, deskripsi, dan detail lainnya.

Berikut adalah penjelasan singkat tentang masing-masing fungsi:

- Fungsi mint: Digunakan untuk menciptakan token baru dengan URI (*Uniform Resource Identifier*) tertentu. Setiap token yang dibuat memiliki ID token yang unik. Fungsi ini meningkatkan ID token, membuat token baru dengan pemilik yang memanggil fungsi, dan menetapkan URI token untuk menyimpan informasi tambahan tentang token.
- Fungsi totalSupply: Digunakan untuk me-return jumlah total token yang telah dibuat oleh *contract*. Ini memungkinkan pengguna untuk mengetahui berapa banyak token yang ada dalam sirkulasi.

B. Implementasi Escrow Contract

Implementasi *escrow contract* dibagi menjadi beberapa bagian sesuai dengan fungsinya, yaitu:

- Interface IERC721

```
interface IERC721 {
    function transferFrom(
        address _from,
        address _to,
        uint256 _id
    ) external;
}
```

Bagian ini merupakan interface minimal untuk *escrow contract*. Dengan interface ini, *escrow contract* yang dibuat dipastikan memiliki standar ERC-721.

- *Modifier*

```
modifier onlyBuyer(uint256 _nftID) { ... }
modifier onlySeller() { ... }
modifier onlyInspector() { ... }
```

Fungsi *modifier* dalam Solidity adalah blok kode yang digunakan untuk mengubah perilaku fungsi lainnya di dalam *smart contract*. *Modifiers* memungkinkan penambahan pemeriksaan atau logika sebelum atau sesudah eksekusi fungsi utama tanpa harus menulis ulang kode tersebut dalam setiap fungsi yang memerlukannya. Dalam konteks implementasi ini, fungsi ini digunakan untuk membatasi akses ke fungsi-fungsi tertentu hanya untuk *buyer*, *seller*, atau *inspector*.

- *List*

```
function list(
    uint256 _nftID,
    address _buyer,
    uint256 _purchasePrice,
    uint256 _escrowAmount
) public payable onlySeller {
    // Transfer NFT dari penjual ke kontrak ini
    IERC721(nftAddress).transferFrom(msg.sender, address(this), _nftID);

    isListed[_nftID] = true;
    purchasePrice[_nftID] = _purchasePrice;
    escrowAmount[_nftID] = _escrowAmount;
    buyer[_nftID] = _buyer;
}
```

Fungsi ini digunakan oleh penjual untuk mencantumkan properti *real estate* (dalam bentuk NFT) untuk dijual. NFT akan ditransfer dari penjual ke *contract* ini, dan beberapa informasi seperti harga pembelian dan adalah jumlah uang yang harus ditempatkan oleh pembeli sebagai deposit awal (escrowAmount).

- *Deposit earnest*

```
function depositEarnest(uint256 _nftID) public payable onlyBuyer(_nftID) {
    require(msg.value >= escrowAmount[_nftID]);
}
```

Dalam fungsi ini, msg.value harus sama atau lebih besar dari escrowAmount yang ditetapkan untuk nftID tertentu. Modifier onlyBuyer memastikan bahwa hanya pembeli yang sah yang dapat memanggil fungsi ini. Fungsi ini memungkinkan pembeli harus mendepositokan *earnest money* (uang muka) ke dalam *escrow contract*.

- *Update inspection status*

```
function updateInspectionStatus(uint256 _nftID, bool _passed)
    public
    onlyInspector
{
    inspectionPassed[_nftID] = _passed;
}
```

Fungsi ini memungkinkan inspektur untuk memperbarui status inspeksi dari NFT.

- *Approve sale*

```
function approveSale(uint256 _nftID) public {
    approval[_nftID][msg.sender] = true;
}
```

Dalam *escrow contract*, penjualan *tokenized real estate* membutuhkan persetujuan dari beberapa pihak seperti pembeli, penjual, *lender*, *inspector*, dll. Fungsi ini memungkinkan pihak-pihak yang terlibat dalam transaksi untuk menyetujui penjualan sebuah *real estate*. Mapping *approval* menyimpan status persetujuan untuk setiap nftID dari setiap alamat (*address*). Ketika fungsi ini dipanggil, *address* pengirim (msg.sender) memberikan persetujuannya untuk penjualan *real estate* dengan ID yang ditentukan. Persetujuan ini dicatat dengan menyetelnya ke *true*.

- *Finalize sale*

```
function finalizeSale(uint256 _nftID) public {
    require(inspectionPassed[_nftID]);
    require(approval[_nftID][buyer[_nftID]]);
    require(approval[_nftID][seller]);
    require(approval[_nftID][lender]);
    require(address(this).balance >= purchasePrice[_nftID]);

    isListed[_nftID] = false;

    (bool success, ) = payable(seller).call{value: address(this).balance}("");
    require(success);

    IERC721(nftAddress).transferFrom(address(this), buyer[_nftID], _nftID);
}
```

Fungsi ini digunakan untuk menyelesaikan penjualan *real estate* setelah semua persyaratan yang diperlukan telah dipenuhi. Persyaratan yang dimaksud mencakup: Inspeksi telah disetujui, semua pihak (pembeli, penjual, dan pemberi pinjaman) telah menyetujui penjualan, dan *contract* memiliki cukup dana untuk membayar penjual. Setelah melakukan finalisasi, NFT dipindahkan kepemilikannya ke pembeli, dan kemudian ditandai tidak lagi terdaftar untuk dijual.

- *Cancel sale*

```
function cancelSale(uint256 _nftID) public {
    if (inspectionPassed[_nftID] == false) {
        payable(buyer[_nftID]).transfer(address(this).balance);
    } else {
        payable(seller).transfer(address(this).balance);
    }
}
```

Fungsi ini menangani pembatalan penjualan *tokenized real estate*. Fungsi ini menentukan tindakan yang harus diambil terkait dengan dana *escrow* ketika penjualan dibatalkan berdasarkan status inspeksi. Jika inspeksi tidak lulus, dana yang disimpan dalam *contract* (*escrow*) akan dikembalikan kepada *buyer*. Namun, jika inspeksi lulus, dana yang disimpan dalam *contract* (*escrow*) akan dikirimkan kepada penjual *seller*.

- *Receive*

```
receive() external payable {}
```

Fungsi ini memungkinkan *contract* untuk menerima pembayaran dalam bentuk Ethereum. Fungsi *receive* adalah fungsi *fallback* yang otomatis dipanggil ketika *contract* menerima Ethereum tanpa memanggil fungsi tertentu.

- *Get balance*

```
function getBalance() public view returns (uint256) {
    return address(this).balance;
}
```

Fungsi ini digunakan untuk memeriksa berapa banyak Ethereum yang disimpan dalam *contract* pada suatu waktu tertentu. Hal ini berguna untuk mengetahui jumlah dana yang tersedia dalam *contract*, misalnya untuk audit, verifikasi sebelum melakukan tindakan tertentu, atau sekadar untuk memantau dana yang ada.

V. Testing & Deployment Script

A. *Unit Testing Script*

Berikut adalah *unit testing script* untuk *escrow contract*. Skrip ini menggunakan Chai untuk asertasi dan Hardhat untuk menyediakan *testing environment*.

```
const { expect } = require('chai');
const { ethers } = require('hardhat');

const tokens = (n) => {
    return ethers.utils.parseUnits(n.toString(),
'ether')
}

describe('Escrow', () => {
    let buyer, seller, inspector, lender
    let realEstate, escrow

    beforeEach(async () => {
        // Setup accounts
        [buyer, seller, inspector, lender] = await
ethers.getSigners()

        // Deploy Real Estate
        const RealEstate = await
ethers.getContractFactory('RealEstate')
        realEstate = await RealEstate.deploy()

        // Mint
        let transaction = await
realEstate.connect(seller).mint("https://ipfs.io/i
pfs/QmTudSYeM7mz3PkYEWXWqPjomRPHogcMFSq7XAvsvsgAPS
")
        await transaction.wait()

        // Deploy Escrow
        const Escrow = await
ethers.getContractFactory('Escrow')
        escrow = await Escrow.deploy(
            realEstate.address,
            seller.address,
            inspector.address,
            lender.address
        )

        // Approve Property
        transaction = await
realEstate.connect(seller).approve(escrow.address,
1)
        await transaction.wait()

        // List Property
        transaction = await
escrow.connect(seller).list(1, buyer.address,
tokens(10), tokens(5))
        await transaction.wait()
    })

    describe('Deployment', () => {
        it('Returns NFT address', async () => {
            const result = await
escrow.nftAddress()

expect(result).to.be.equal(realEstate.address)
        })

        it('Returns seller', async () => {
            const result = await escrow.seller()

expect(result).to.be.equal(seller.address)
        })

        it('Returns inspector', async () => {
            const result = await
escrow.inspector()

expect(result).to.be.equal(inspector.address)
        })

        it('Returns lender', async () => {
            const result = await escrow.lender()

expect(result).to.be.equal(lender.address)
        })
    })

    describe('Listing', () => {
        it('Updates as listed', async () => {
            const result = await
escrow.isListed(1)
            expect(result).to.be.equal(true)
        })

        it('Returns buyer', async () => {
            const result = await escrow.buyer(1)

expect(result).to.be.equal(buyer.address)
        })

        it('Returns purchase price', async () => {
            const result = await
escrow.purchasePrice(1)
            expect(result).to.be.equal(tokens(10))
        })

        it('Returns escrow amount', async () => {
            const result = await
escrow.escrowAmount(1)
            expect(result).to.be.equal(tokens(5))
        })

        it('Updates ownership', async () => {
            expect(await
realEstate.ownerOf(1)).to.be.equal(escrow.address)
        })
    })

    describe('Deposits', () => {
        beforeEach(async () => {
            const transaction = await
escrow.connect(buyer).depositEarnest(1, { value:
tokens(5) })
            await transaction.wait()
        })

        it('Updates contract balance', async () =>
{
            const result = await
escrow.getBalance()
            expect(result).to.be.equal(tokens(5))
```

```
        })
    })

    describe('Inspection', () => {
        beforeEach(async () => {
            const transaction = await
escrow.connect(inspector).updateInspectionStatus(1
, true)
            await transaction.wait()
        })

        it('Updates inspection status', async ()
=> {
            const result = await
escrow.inspectionPassed(1)
            expect(result).to.be.equal(true)
        })
    })

    describe('Approval', () => {
        beforeEach(async () => {
            let transaction = await
escrow.connect(buyer).approveSale(1)
            await transaction.wait()

            transaction = await
escrow.connect(seller).approveSale(1)
            await transaction.wait()

            transaction = await
escrow.connect(lender).approveSale(1)
            await transaction.wait()
        })

        it('Updates approval status', async () =>
{
            expect(await escrow.approval(1,
buyer.address)).to.be.equal(true)
            expect(await escrow.approval(1,
seller.address)).to.be.equal(true)
            expect(await escrow.approval(1,
lender.address)).to.be.equal(true)
        })
    })

    describe('Sale', () => {
        beforeEach(async () => {
            let transaction = await
escrow.connect(buyer).depositEarnest(1, { value:
tokens(5) })
            await transaction.wait()

            transaction = await
escrow.connect(inspector).updateInspectionStatus(1
, true)
            await transaction.wait()

            transaction = await
escrow.connect(buyer).approveSale(1)
            await transaction.wait()

            transaction = await
escrow.connect(seller).approveSale(1)
            await transaction.wait()

            transaction = await
escrow.connect(lender).approveSale(1)
            await transaction.wait()

            await lender.sendTransaction({ to:
escrow.address, value: tokens(5) })

            transaction = await
escrow.connect(seller).finalizeSale(1)
            await transaction.wait()
        })

        it('Updates ownership', async () => {
            expect(await
realEstate.ownerOf(1)).to.be.equal(buyer.address)
        })

        it('Updates balance', async () => {
            expect(await
escrow.getBalance()).to.be.equal(0)
        })
    })
})
```

Untuk melakukan *unit test*, gunakan command ‘npx hardhat test’. Berikut adalah tampilan setelah menjalankan *unit testing script*.

```
Escrow
  Deployment
    ✓ Returns NFT address (38ms)
    ✓ Returns seller
    ✓ Returns inspector
    ✓ Returns lender
  Listing
    ✓ Updates as listed
    ✓ Returns buyer
    ✓ Returns purchase price
    ✓ Returns escrow amount
    ✓ Updates ownership
  Deposits
    ✓ Updates contract balance
  Inspection
    ✓ Updates inspection status
  Approval
    ✓ Updates approval status
  Sale
    ✓ Updates ownership
    ✓ Updates balance


14 passing (9s)
```

B. Deployment Script

Berikut adalah *deployment script* menggunakan Hardhat, *framework* untuk pengembangan *smart contract* di Ethereum.

```
const hre = require("hardhat");

const tokens = (n) => {
  return ethers.utils.parseUnits(n.toString(),
"ether");
};
```

```
async function main() {
  // Setup accounts
  const [buyer, seller, inspector, lender] = await
ethers.getSigners();

  // Deploy Real Estate
  const RealEstate = await
ethers.getContractFactory("RealEstate");
  const realEstate = await RealEstate.deploy();
  await realEstate.deployed();

  console.log(`Deployed Real Estate Contract at:
${realEstate.address}`);
  console.log(`Minting 3 properties...\n`);

  for (let i = 0; i < 3; i++) {
    const transaction = await realEstate
      .connect(seller)
      .mint(

`https://ipfs.io/ipfs/QmQVcpsjrA6cr1iJjZAodYwmPekY
gbnXGo4DFubJiLc2EB/${
          i + 1
        }.json`
      );
    await transaction.wait();
  }

  // Deploy Escrow
  const Escrow = await
ethers.getContractFactory("Escrow");
  const escrow = await Escrow.deploy(
    realEstate.address,
    seller.address,
    inspector.address,
    lender.address
  );
  await escrow.deployed();

  console.log(`Deployed Escrow Contract at:
${escrow.address}`);
  console.log(`Listing 3 properties...\n`);

  let transaction;
  for (let i = 0; i < 3; i++) {
    // Approve properties...
    transaction = await realEstate
      .connect(seller)
      .approve(escrow.address, i + 1);
    await transaction.wait();
  }

  // Listing properties...
  transaction = await escrow
    .connect(seller)
    .list(1, buyer.address, tokens(20),
tokens(10));
  await transaction.wait();

  transaction = await escrow
    .connect(seller)
    .list(2, buyer.address, tokens(15),
tokens(5));
  await transaction.wait();

  transaction = await escrow
    .connect(seller)
    .list(3, buyer.address, tokens(10),
tokens(5));
  await transaction.wait();

  console.log(`Finished.`);
}

main().catch((error) => {
  console.error(error);
  process.exitCode = 1;
});
```

Untuk melakukan *deployment*, gunakan command ‘npx hardhat run ./scripts/deploy.js --network localhost’. Berikut adalah tampilan setelah menjalankan *deployment script*.

```
Deployed Real Estate Contract at: 0x5FbDB2315678afecb367f032d93F642f64180aa3
Minting 3 properties...

Deployed Escrow Contract at: 0xe7f1725E7734CE288F8367e1Bb143E90bb3F0512
Listing 3 properties...

Finished.
```

VI. Penutup

Penerapan teknologi *blockchain* dalam transaksi *real estate* memberikan berbagai keuntungan signifikan yang dapat mengatasi banyak masalah dalam industri ini. Dengan sifat desentralisasi dan transparansinya, *blockchain* mampu mengurangi risiko penipuan, meningkatkan efisiensi, serta mempercepat proses transaksi. Implementasi *smart contracts* dapat mengurangi ketergantungan pada perantara seperti notaris dan agen properti. Selain itu, penggunaan NFT (*Non-Fungible Tokens*) menjamin kepastian hukum dan keamanan kepemilikan properti serta penggunaan IPFS (*InterPlanetary File System*) untuk penyimpanan metadata menjamin integritas dan aksesibilitas data secara permanen dan terdesentralisasi.

Untuk mengatasi tantangan teknis dan regulasi dalam implementasi *blockchain* di sektor *real estate*, diperlukan penelitian lebih lanjut dan pengembangan teknologi yang berkelanjutan. Regulasi yang jelas dan kerangka hukum yang mendukung sangat penting untuk menjamin kepastian hukum dan perlindungan konsumen. Tak hanya itu, pemerintah dan lembaga terkait harus berkolaborasi dalam mengembangkan regulasi yang tepat. Edukasi dan sosialisasi tentang teknologi *blockchain* kepada agen properti, notaris, dan konsumen juga tak kalah penting karena akan mempercepat adopsi. Dengan ini, diharapkan teknologi *blockchain* dapat memberikan kontribusi yang signifikan dalam meningkatkan efisiensi, keamanan, dan transparansi dalam transaksi *real estate*, serta mendorong pertumbuhan ekonomi dan inovasi di sektor ini.

Source Code Link at Github

https://github.com/ceavinrufus/estate-net

Video Link at Youtube

https://youtu.be/L7lREVNlRhI

References

[1] Tostevin, Paul. (2021, September 21). *Value of global real estate rises 5% to $326.5 trillion*. Savills. Retrieved 08 June, 2024, from .https://www.savills.com/insight-and-opinion/savills-news/319145/value-of-global-real-estate-rises-5--to-$326.5-trillion

[2] Bogwasi, T. *What is Real Estate?*. Brimco. Retrieved 08 June, 2024, from https://www.brimco.io/realestate/what-is-real-estate/

[3] International Business Machines. *What is Blockchain?*. https://www.ibm.com/topics/blockchain

[4] Munir, Rinaldi. (2024). Penggunaan Kriptografi di dalam Blockchain. Institut Teknologi Bandung.

[5] Amazon Web Services. Apa itu Teknologi *Blockchain*?. https://aws.amazon.com/id/what-is/blockchain/

[6] Chen, James. (2023, May 31). *Real Estate: Definition, Types, How to Invest in It*. Investopedia. Retrieved 09 June, 2024, from https://www.investopedia.com/terms/r/realestate.asp

[7] Asadov, A.I., Ibrahim, M.H. & Yildirim, R. (2023). *Impact of House Price on Economic Stability: Some Lessons from OECD Countries*. J Real Estate Finan Econ. https://doi.org/10.1007/s11146-023-09945-0

[8] Kattan, Danny. (2020, December 9). *The Impact Of Technology On Real Estate*. Forbes. Retrieved 10 June, 2024, from https://www.forbes.com/sites/forbesrealestatecouncil/2020/12/09/the-impact-of-technology-on-real-estate/

[9] Ethereum. *What is Ethereum?*. https://ethereum.org/en/what-is-ethereum/

[10] K, Amira. Mengenal Apa itu NFT dan Bagaimana Cara Kerjanya. Retrieved 10 June, 2024, from https://www.gramedia.com/literasi/mengenal-nft/

[11] Conty, Robin. (2024, May 10). *What Is An NFT? Non-Fungible Tokens Explained*. Retrieved 10 June, 2024, from https://www.forbes.com/advisor/investing/cryptocurrency/nft-non-fungible-token/

[12] International Business Machines. *What are smart contracts on blockchain?*. https://www.ibm.com/topics/smart-contracts

[13] IPFS. *What is IPFS*. https://docs.ipfs.tech/concepts/what-is-ipfs/#defining-ipfs

[14] Ethereum. (2023) *ERC-721 Non-Fungible Token Standard.*. https://ethereum.org/en/developers/docs/standards/tokens/erc-721/

[15] Chen, James. (2021). *What Is an Escrow Agreement? How It Works, Uses, and Types*. Investopedia. Retrieved 11 June, 2024, from https://www.investopedia.com/terms/e/escrowagreement.asp

[16] The Investopedia Team. (2024, June 08). *Cryptocurrency Wallet: What It Is, How It Works, Types, and Security*. Investopedia. Retrieved 11 June, 2024, from https://www.investopedia.com/terms/b/bitcoin-wallet.asp

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Jakarta, 12 Juni 2024

Ceavin Rufus De Prayer Purba (18221162)