TUGAS

BLOCKCHAIN

Dosen Pengampu:

Ade Kurniawan, S.Pd., M.Pd.T

Disusun Oleh:

Galant Anefsyah Pratama (22346031)

PRODI INFORMATIKA

FAKULTAS ELEKTRONIKA

UNIVERSITAS NEGERI PADANG

TAHUN AJARAN 2025/2025

Ringkasan Materi

A. Smart Contract

1. Pengertian

   Smart contract adalah program komputer yang berjalan di atas blockchain Ethereum dan secara otomatis mengeksekusi perintah sesuai dengan aturan yang telah ditentukan. Smart contract digunakan untuk menghilangkan peran pihak ketiga dalam transaksi digital dan memberikan kepastian bahwa kontrak akan berjalan sesuai ketentuan yang ditetapkan.

2. Karakteristik Smart Contract
   a. **Otomatis** – Setelah dikodekan dan diterapkan di blockchain, kontrak ini akan berjalan secara otomatis tanpa perlu campur tangan manusia.
   b. **Desentralisasi** – Smart contract berjalan di jaringan Ethereum yang tidak dikontrol oleh satu entitas tertentu.
   c. **Keamanan** – Data yang tersimpan dalam smart contract tidak dapat diubah setelah dikonfirmasi di blockchain, sehingga mengurangi risiko manipulasi.
   d. **Transparan** – Semua transaksi dan eksekusi kontrak dapat dilihat oleh semua pengguna jaringan Ethereum.
   e. **Efisien** – Mengurangi biaya administrasi dan waktu yang dibutuhkan dalam proses hukum tradisional.

3. Contoh Penggunaan Smart Contract
   a. **Keuangan Terdesentralisasi (DeFi)** – Digunakan untuk membuat layanan pinjaman, staking, dan pertukaran aset digital.
   b. **NFT (Non-Fungible Token)** – Digunakan untuk menciptakan aset digital unik seperti karya seni digital dan koleksi game.
   c. **Manajemen Rantai Pasok** – Memastikan keaslian dan kelancaran distribusi barang dalam suatu ekosistem bisnis.

B. GAS

1. Pengertian

   GAS adalah unit biaya dalam jaringan Ethereum yang diperlukan untuk melakukan transaksi atau menjalankan smart contract. Biaya GAS ini dibutuhkan untuk membayar validator atau miner yang memproses transaksi dalam jaringan.

2. Fungsi GAS dalam Ethereum
   a. **Mencegah spam** – Dengan menerapkan biaya transaksi, Ethereum menghindari penyalahgunaan jaringan.
   b. **Menentukan prioritas transaksi** – Semakin tinggi biaya GAS yang ditawarkan, semakin cepat transaksi diproses oleh validator.
   c. **Mengompensasi validator** – Validator mendapatkan reward atas usaha mereka dalam mengamankan jaringan dan memproses transaksi.

3. Komponen GAS
   a. **Gas Limit** – Jumlah maksimum GAS yang bersedia dibayar pengguna untuk suatu transaksi.
   b. **Gas Price** – Harga yang bersedia dibayar per unit GAS (diukur dalam Gwei, 1 Gwei = 10^9 Wei).
   c. **Total Gas Fee** – Biaya keseluruhan transaksi dihitung dengan rumus:

   $$\text{Total Gas Fee} = \text{Gas Used} \times \text{Gas Price}$$

   d. **Base Fee & Priority Fee** –
      1) **Base Fee** adalah biaya dasar transaksi yang disesuaikan berdasarkan kepadatan jaringan.
      2) **Priority Fee** adalah tip tambahan untuk mempercepat transaksi.

4. Contoh Perhitungan GAS

   Misalkan Gas Price adalah **50 Gwei** dan Gas Used adalah **21.000 unit**, maka:

   $$\text{Total Gas Fee} = 21.000 \times 50 = 1.050.000 \text{ Gwei} = 0.00105 \text{ ETH}$$

C. Ether (ETH)

1. Pengertian

   Ether (ETH) adalah mata uang kripto utama dalam jaringan Ethereum. ETH digunakan sebagai alat pembayaran biaya transaksi, eksekusi smart contract, serta berbagai layanan di dalam ekosistem Ethereum.

2. Fungsi Ether (ETH)
    a. **Membayar biaya transaksi (GAS Fee)** dalam jaringan Ethereum.
    b. **Berpartisipasi dalam staking Ethereum 2.0** untuk membantu mengamankan jaringan.
    c. **Sebagai alat tukar dalam berbagai aplikasi terdesentralisasi (DApps).**
    d. **Digunakan sebagai aset investasi** karena nilainya dapat meningkat seiring waktu.

D. Account

1. Pengertian

   Account dalam Ethereum adalah entitas yang dapat mengirim transaksi, menerima dana, dan berinteraksi dengan smart contract.
2. Jenis Account
    a. **Externally Owned Account (EOA)** – Akun yang dikendalikan oleh pengguna dengan private key.
    b. **Contract Account** – Akun yang dikendalikan oleh kode smart contract tanpa memerlukan private key.
3. Komponen Account
    a. **Address** – Alamat unik yang digunakan untuk menerima dan mengirim ETH.
    b. **Balance** – Jumlah ETH yang dimiliki dalam akun.
    c. **Nonce** – Jumlah transaksi yang telah dilakukan oleh akun untuk mencegah replay attack.
4. Contoh Address Ethereum

   Sebuah address Ethereum memiliki format seperti:

   0x742d35Cc6634C0532925a3b844Bc454e4438f44e

E. Transaction

1. Pengertian

   Transaction adalah perintah yang dikirim ke blockchain Ethereum untuk memindahkan ETH atau mengeksekusi fungsi dalam smart contract.
2. Jenis Transaksi di Ethereum
    a. **Transfer ETH** – Mengirim ETH dari satu akun ke akun lain.

b. **Interaksi dengan Smart Contract** – Memanggil fungsi dalam smart contract, seperti membeli NFT atau menggunakan layanan DeFi.
c. **Deploy Smart Contract** – Mengunggah kode smart contract ke jaringan Ethereum.

3. Struktur Transaksi
   a. **Nonce** – Menghindari duplikasi transaksi.
   b. **To** – Alamat penerima transaksi.
   c. **Value** – Jumlah ETH yang dikirim.
   d. **Gas Limit & Gas Price** – Menentukan biaya yang dibayarkan untuk transaksi.
   e. **Data** – Informasi tambahan (terutama untuk smart contract).
   f. **Signature** – Tanda tangan digital sebagai bukti otentikasi transaksi.
4. Proses Transaksi di Ethereum
   a. Pengguna mengirim transaksi ke jaringan.
   b. Validator memverifikasi transaksi dan memprosesnya.
   c. Transaksi dikonfirmasi dan ditambahkan ke blockchain.

F. Smart Contract dengan menngunakan Remix IDE

1. Alat dan Bahan

a. Remix IDE (https://remix.ethereum.org)
b. Metamask Wallet (Testnet Sepolia)

2. Langkah – Langkah

a. Buka pada browser anda lalu ketiikan pencarian Remix IDE.
b. Buat file baru: SimpleContract.sol.
c. Salin kode berikut:

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

contract SimpleContract {
    string public message;
    address public owner;

    event MessageUpdated(string oldMessage, string newMessage);
```

```
    modifier onlyOwner() {
        require(msg.sender == owner, "Only the owner can update
the message");
        _;
    }

    constructor(string memory _message) {
        message = _message;
        owner = msg.sender;
    }

    function updateMessage(string memory _newMessage) public
onlyOwner {
        emit MessageUpdated(message, _newMessage);
        message = _newMessage;
    }
}
```

d. Compile file.

e. Pilih Environment: Remix VM (Cancun).

f. Ketikkan perintah pada string_message, misalnya; 1

g. Isi address wallet pada Metamask anda

h. Klik Deploy

i. Hasil akhir sebagai berikut: