JURNAL INFORMATIKA DAN TEKONOLOGI KOMPUTER
Halaman Jurnal: https://journal.amikveteran.ac.id/index.php/jitek
Halaman UTAMA Jurnal : https://journal.amikveteran.ac.id/index.php

DOI : https://doi.org/10.55606/jitek.v3i3.2001

PEMBUATAN SMARTCONTRACT APLIKASI FUNDRAISING BERBASIS BLOCKCHAIN

Budianto [a*], Retno Mumpuni [b], Henni Endah Wahanani [c]
[a] Fakultas Ilmu Komputer / Program Studi Informatika, budiantojang@gmail.com,
Universitas Pembangunan Nasional "Veteran" Jawa Timur
[b] Fakultas Ilmu Komputer / Program Studi Informatika, retnomumpuni.if@upnjatim.ac.id,
Universitas Pembangunan Nasional "Veteran" Jawa Timur
[c] Fakultas Ilmu Komputer / Program Studi Informatika, henniendah.if@upnjatim.ac.id,
Universitas Pembangunan Nasional "Veteran" Jawa Timur
[*] Correspondence

ABSTRACT

Fundraising activities have become a common method to achieve various objectives, including charitable activities and gathering funds from investors for companies. However, cases of fund misuse often occur, leading to inappropriate use of funds or even unfair loss. Therefore, an innovative approach is needed to enhance trust and transparency in the fundraising process. In this study, the use of smart contracts on a blockchain network will be employed as a solution to ensure the security and integrity of each transaction in fundraising activities. By utilizing smart contracts, all transactions in fundraising activities can be verified by all users on the blockchain network. Transaction data recorded on the blockchain is permanent and tamper-proof, preventing any manipulation or fund misuse. Leveraging smart contracts and blockchain technology, fundraising activities can be conducted more efficiently, transparently, and securely. This approach helps improve trust among involved parties and reduces the risk of fund misuse

Keywords: *smartcontract*, *blockchain*, *fund*raising, *immutable*, *transactions*

ABSTRAK

Kegiatan *fundraising* telah menjadi metode umum dalam mencapai berbagai tujuan, termasuk kegiatan amal dan pengumpulan dana dari investor untuk perusahaan. Namun, sering terjadi kasus penyalahgunaan dana yang mengakibatkan penggunaan dana yang yang tidak sesuai dengan tujuan awal atau bahkan hilang secara tidak adil. Oleh karena itu, diperlukan pendekatan inovatif untuk mengingkatkan kepercayaan dan transparansi dalam proses penggalangan dana. Dalam penelitian ini, penggunaan *smartcontract* dalam jaringan *blockchain* akan digunakan sebagai solusi untuk menjaga keamanan dan integritas setiap transaksi dalam kegiatan *fundraising*. Melalui pengunaan *smartcontract*, semua transaksi dalam kegiatan *fundraising* dapat diverifikasi oleh seluruh pengguna dalam jaringan *blockchain*. Data transaksi yang dicatat dalam *blockchain* bersifat permanen dan tidak dapat diubah, sehingga mencegah adanya manipulasi atau penyalahgunaan dana. Dengan memanfaatkan *smartcontract* dan teknologi *blockchain*, kegiatan *fundraising* dapat dilakukan secara lebih efisien, transparan, dan aman. Pendekatan ini membantu meningkatkan kepercayaan para pihak yang terlibat, serta mengurangi resiko penyalahgunaan dana.

Kata Kunci: *smartcontract*, *blockchain*, penggalangan dana, *immutable*, transaksi

1. PENDAHULUAN

Blockchain [1] merupakan teknologi menggunakan desentralisasi dan enkripsi sebagai konsep dasarnya. Desentralisasi [2] memungkinkan semua pengguna dalam jaringan *blockchain* untuk melakukan verifikasi dan menjadi bagian dari pemeriksa setiap transaksi, yang dapat diakses oleh seluruh perangkat di dunia. Sementara itu, enkripsi menggunakan konsep *asymetric encryption* untuk memastikan integritas dan

Received juli 19, 2023; Revised Agustus 26, 2023; Accepted Oktober 30, 2023

keotentikan data. Data yang dikirimkan melalui *blockchain* akan ditambahkan dengan *private key* didalamnya sehingga data tersebut tidak dapat dipalsukan tanpa memiliki akses terhadap *private key* tersebut.

Dalam *blockchain* terdapat teknologi yang bernama *smartcontract* [4] yang memungkinkan transaksi didalam *blockchain* menjadi lebih fleksibel dan dapat mengandung serta menggunakan algoritma-algoritma didalamnya yang memungkinkan automasi, eksekusi perintah, dan integrasi proses bisnis melalui kode yang terkandung didalamnya. Dengan memanfaatkan *smartcontract*, semua keunggulan dari *blockchain* dapat dimanfaatkan, termasuk dalam penanganan proses bisnis yang kompleks seperti transaksi [5] dana.

Dengan menerapkan pendekatan tersebut, dapat dibangun sebuah sistem dimana proses transaksi dalam sebuah aplikasi *fundraising* dapat dijalankan secara otomatis [6] ketika *smartcontract* dipanggil dan proses pencatatan secara otomatis dalam jaringan blockchain. Transaksi yang tercatat dalam blockchain tidak dapat dimanipulasi karena data dalam *blockchain* bersifat *immutable* [7], artinya setelah data masuk kedalaam sebuah blok, data tersebut akan ada dan tersimpan secara permanen [8] dan hanya dapat diubah melalui transaksi yang dilakukan di masa depan.

2. TINJAUAN PUSTAKA

2.1. Desentralisasi

Desentralisasi adalah pembagian proses kekuasaan terpusat secara keseluruhan menjadi terpisah-pisah. Teknologi *blockchain* adalah teknologi dasar yang memungkinkan desentralisasi dan memberikan kesempatan kepada semua pengguna untuk menjadi bagian dari validator atau node untuk transaksi di masa depan. Desentralisasi merujuk pada konsep yang digunakan untuk menggambarkan sistem yang tidak memiliki satu entitas sentral yang mengendalikan atau memiliki semua data dan informasi dalam jaringan. Sebaliknya, informasi dan keputusan dibagi secara merata di antara semua pengguna dan *node* dalam jaringan.

2.2. Blockchain

Blockchain merupakan jaringan terdistribusi yang didasarkan pada prinsip desentralisasi dan enkripsi. Prinsip desentralisasi memungkinkan semua pengguna dalam jaringan untuk melakukan verifikasi transaksi dan menjadi bagian dari proses pemeriksaan. Hal ini menghasilkan sistem yang transparan, di mana semua transaksi dapat diakses oleh seluruh perangkat di dunia.

Selain itu, *blockchain* juga menggunakan enkripsi untuk memastikan keamanan dan integritas data. Konsep enkripsi menggunakan metode *asymetric encryption*, di mana data yang dikirimkan melalui blockchain ditambahkan dengan *private key* yang hanya dapat diakses oleh pemiliknya. Hal ini menjaga data agar tidak dapat dipalsukan atau diubah tanpa memiliki akses terhadap *private key* yang tepat.

Salah satu aspek penting dari teknologi *blockchain* adalah adanya *smart contract*. *Smart contract* memungkinkan transaksi di dalam *blockchain* menjadi lebih fleksibel dan dapat mengandung algoritma-algoritma yang memungkinkan otomasi, eksekusi perintah, dan integrasi proses bisnis melalui kode yang terkandung dalamnya. Dengan memanfaatkan *smart contract*, *blockchain* dapat digunakan dalam berbagai bidang, termasuk dalam penanganan proses bisnis yang kompleks seperti transaksi keuangan.

2.3. Enkripsi Asimetris

Enkripsi asimetris, juga dikenal sebagai kriptografi kunci publik, adalah konsep fundamental dalam kriptografi modern yang memberikan cara yang aman untuk mentransmisikan dan menyimpan informasi yang sensitif. Berbeda dengan enkripsi simetris yang menggunakan satu kunci untuk enkripsi dan dekripsi, enkripsi asimetris menggunakan sepasang kunci yang saling terkait secara matematis: kunci publik dan kunci privat.

Keunikan enkripsi asimetris terletak pada kemampuannya untuk menjaga kerahasiaan pesan dan mengamankan transmisi data tanpa perlu menyampaikan kunci privat secara terbuka. Ini sangat penting dalam lingkungan komunikasi yang tidak terpercaya, seperti jaringan internet. Dengan menggunakan kriptografi asimetris, penerima pesan dapat memverifikasi keaslian pesan dengan menggunakan kunci publik yang sesuai, sedangkan hanya pemilik kunci privat yang dapat mendekripsinya. Dengan demikian, enkripsi asimetris memungkinkan pertukaran pesan aman dan otentikasi tanpa harus mengungkapkan kunci rahasia kepada pihak lain.

2.4. Smart Contract

Smart contract adalah program komputer yang memiliki sifat *self-verifying*, *self-executing*, dan *tamper-resistant*. Konsep *smart contract* pertama kali diusulkan oleh Nick Szabo pada tahun 1994. *Smart contract* memungkinkan eksekusi kode tanpa pihak ketiga. *Smart contract* terdiri dari nilai, alamat, fungsi, dan *state*. *Smart contract* menerima transaksi sebagai input, mengeksekusi kode yang sesuai, dan memicu *output events*. Bergantung pada implementasi logika fungsi, *state* dapat berubah.

Sejak tahun 2008, ketika teknologi *blockchain* muncul melalui *cryptocurrency* Bitcoin, integrasi *smart contract* dengan teknologi blockchain menjadi fokus untuk dikembangkan karena memberikan transaksi *peer-to-peer* dan *database* dapat dipelihara secara publik dengan cara yang aman dalam lingkungan yang dapat dipercaya. *Smart contract* dapat dilacak dan tidak dapat dibatalkan. Semua informasi transaksi terdapat dalam smart contract dan dieksekusi secara otomatis.

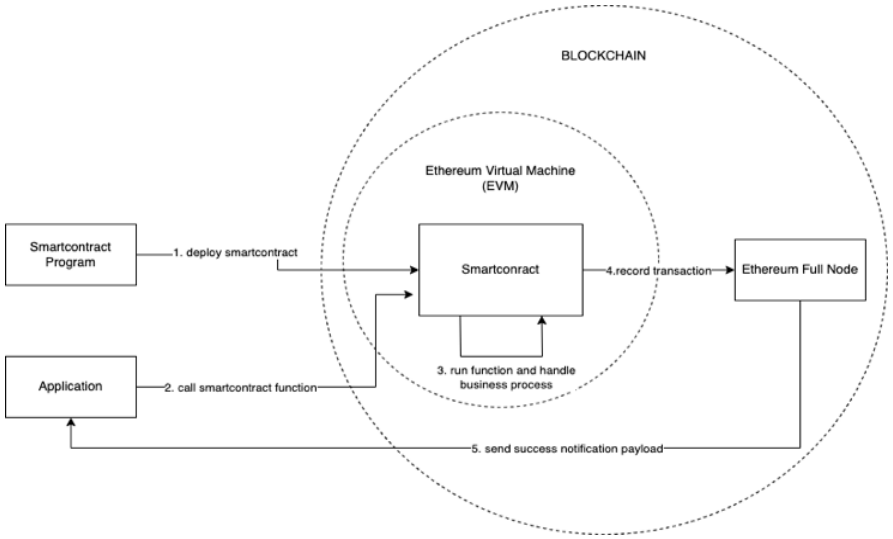
2.5. Solidity

Solidity adalah bahasa pemrograman yang digunakan untuk mengembangkan aplikasi *blockchain* pada platform Ethereum. Solidity dirancang khusus untuk memungkinkan pengembang aplikasi *blockchain* menulis *smart contract* pada jaringan Ethereum.

3. METODOLOGI PENELITIAN

3.1. Desain Sistem

Smartcontract yang dibuat dalam jaringan *blockchain* akan berjalan dalam *Ethereum Virtual Machine* (*EVM*) [9]. *EVM* adalah adalah mesin virtual yang berfungsi untuk mengeksekusi program-program yang ada dalam *smartcontract* dan melakukan kompilasi ketika *smartcontract* telah di-*deploy*. *Smartcontract* akan berperan sebagai sistem *backend* bagi aplikasi-aplikasi yang ingin menggunakannya, sehingga sistem tersebut dapat digunakan oleh aplikasi lain yang akan berinteraksi dengan pengguna, seperti yang diilustrasikan pada Gambar 1.



Gambar 1. Desain Sistem

Program *smartcontract* akan di di unggah secara manual ke dalam jaringan *blockchain* yang akan menghasilkan sebuah *address* yang merepresentasikan *smartcontract* tersebut. Jika aplikasi ingin menjalankan fungsi yang ada dalam *smartcontract*, aplikasi tersebut perlu membuat *payload* yang valid sesuai dengan persyaratan dan data yang diperlukan untuk mengirimkan *request* menjalankan fungsi tersebut.

Dalam *EVM*, *smartcontract* akan dieksekusi sesuai dengan fungsi yang dipanggil. Apabila proses tersebut berhasil, transaksi akan dikirimkan ke *Ethereum Full Node* untuk dicatat dalam blok yang didalam sistem *blockcahin*. Setelah itu, aplikasi dapat menerima *payload* notifikasi yang memberikan detail perubahan yang terjadi pada data yang telah didefinisikan ketika fungsi dalam *smartcontract* dipanggil.

3.2. Desain Smartcontract

Dalam *smartcontract*, *struct* dan variabel perlu dibuat untuk menyimpan menyimpan dan mengolah data transaksi, serta fungsi-fungsi juga akan dibuat untuk menangani proses bisnis dengan menggunakan struct dan variabel tersebut. Untuk memberikan representasi yang lebih jelas mengenai pendefinisian *struct*, variabel, dan juga *event*, maka akan direprsentasikan dalam gambar Gambar 2.

```
contract Crowdfunding {
    struct Contributor {
        uint256 contributorId;
        string name;
        address contributorAddress;
    }

    struct Campaign {
        uint256 id;
        uint256 ownerId;
        address owner;
        string title;
        string description;
        string summary;
        string keyPoint;
        uint256 goal;
        uint256 deadline;
        uint256 raised;
        bool closed;
        Contributor[] contributors;
    }

    uint256 public numCampaigns;
    mapping(uint256 => Campaign) public campaigns;

    event CampaignCreated(uint256 id, uint256 ownerId, address ownerAddress, string title, string description,
    string summary, string keyPoint, uint256 goal, uint256 deadline);
    event CampaignFunded(uint256 id, address backer, uint256 goal, uint256 amount, uint256 raised, Contributor
    contributor, Contributor[] contributors);
    event CampaignClosed(uint256 campaignId, uint256 ownerId, uint256 amountRaised);
```

Gambar 2. Definisi Variabel

Dalam *smartcontract*, terdapat dua buah *struct* yang akan digunakan. Pertama, *struct* "Contributor" digunakan untuk menyimpan data kontributor, seperti alamat dan jumlah sumbangan yang diberikan oleh kontributor. *Struct* ini memungkinkan pengelolaan data kontributor secara lebih terorganisir.

Kedua, *struct* "Campaign" digunakan untuk menyimpan data *campaign*, seperti target jumlah dana yang ingin dicapai, nomor *campaign*, judul, dan sebagainya. Data-data kontributor yang telah melakukan pendanaan terhadap sebuah *campaign* akan dimasukkan sebagai referensi dalam *struct campaign* untuk mempermudah pengelolaan data.. Selain itu didefinisikan juga variabel untuk menyimpan data banyaknya *campaign* dan *hash map* untuk melakukan *mapping* data *campaign* yang akan dibuat.

Selain *struct* dan variabel, *event* juga merupakan komponen penting dalam *smartcontract*. *Event* digunakan untuk menghasilkan notifikasi setelah sebuah fungsi berhasil dijalankan, dan memberikan informasi tentang data yang diubah, diolah, atau digunakan sesuai dengan kebutuhan.

Data *event* yang dihasilkan dapat dimanfaatkan oleh aplikasi lain yang berinteraksi dengan *smartcontract*. Aplikasi tersebut dapat melakukan *subscribe* terhadap *event-event* tersebut untuk mendapatkan detail yang diperlukan. Dengan demikian, aplikasi dapat memperoleh informasi *real-time* tentang perubahan atau kejadian penting yang terjadi dalam sistem fundraising.

```
function createCampaign(uint256 ownerId, string memory title, string memory description, string memory summary,
string memory keyPoint, uint256 goal, uint256 deadline) public {
    require(goal > 0, "Goal must be greater than zero.");
    require(deadline > block.timestamp, "Deadline must be in the future.");

    numCampaigns++;

    Campaign storage campaign = campaigns[numCampaigns];
    campaign.id = numCampaigns;
    campaign.ownerId= ownerId;
    campaign.owner = msg.sender;
    campaign.title = title;
    campaign.description = description;
    campaign.summary = summary;
    campaign.keyPoint = keyPoint;
    campaign.goal = goal;
    campaign.deadline = deadline;
    campaign.closed = false;

    emit CampaignCreated(numCampaigns, ownerId, msg.sender, title, description, summary, keyPoint, goal,
    deadline);
}
```

Gambar 3. Fungsi untuk Membuat *Campaign*

Dalam Gambar 3 terdapat sebuah fungsi yang bertujuan untuk membuat sebuah *campaign* baru. Dalam pembuatan sebuah *campaign*, perlu dipastikan bahwa jumlah yang ingin dicapai harus lebih dari 0 agar sebuah *campaign* dapat dianggap valid. Selain itu, batas waktu juga harus lebih dari *timestamp* blok yang memiliki format *unix*. Setelah kedua hal tersebut terpenuhi, maka *campaign* baru akan dibuat dan di *mapping* ke dalam *hashmap* untuk menyimpan data berdasarkan urutan nomor dari *campaign*. Setelah semua proses selesai dan berjalan dengan baik maka *event* akan dijalankan.

```
function fundCampaign(uint256 id, uint256 contributorId,string memory name) public payable {
    require(id > 0 && id <= numCampaigns, "Invalid campaign ID.");
    Campaign storage campaign = campaigns[id];
    require(!campaign.closed, "Campaign is closed.");
    require(campaign.deadline > block.timestamp, "Deadline has passed.");
    require(msg.value > 0, "Must send some ETH.");

    campaign.raised += msg.value;
    campaign.contributors.push(Contributor(contributorId, name, msg.sender));

    Contributor[] memory contributors = getContributors(id);
    emit CampaignFunded(id, msg.sender,campaign.goal, msg.value, campaign.raised, campaign.contributors[campaign.
    contributors.length-1], contributors);
}

function getContributors(uint256 id) public view returns (Contributor[] memory) {
    require(id > 0 && id <= numCampaigns, "Invalid campaign ID.");
    Campaign storage campaign = campaigns[id];
    return campaign.contributors;
}
```

Gambar 4. Fungsi untuk Proses Pendanaan

Proses pendanaan yang ditangani dalam *smartcontract* memerlukan adanya perpindahan dana dari pengguna ke dalam kumpulan data yang akan disimpan dalam *blockchain*. Untuk itu, fungsi yang dibuat harus memiliki keterangan sebagai *payable* untuk mendukung proses perpindahan dana secara eksternal. Dalam fungsi yang digambarkan pada Gambar 4 akan dilakukan berbagai validasi untuk memastikan setiap data yang masuk telah valid dan sesuai dengan ketentuan yang telah dibuat. Setelah itu, akan dilakukan *mapping* dana yang akan mencatat jumlah dana yang telah dikirimkan sesuai dengan data *hash map* dari *campaign* yang diinginkan.

Didalam fungsi tersebut, dilakukan juga pengambilan data data kontributor dari sebuah *campaign* yang nantinya akan digunakan untuk mencatat detail dari kontributor baru yang melakukan transaksi. Sehingga dalam sebuah *campaign* akan terdapat riwayat yang jelas mengenai jumlah dana dan juga nama donatur. Hal tersebut membuat proses bisnis lebih aman dikarenakan data yang telah masuk kedalam *blockchain* tidak dapat diubah sehingga para donatur dapat melakukan pengecekan secara pribadi akan data donasi masing masing. Setelah semua proses selesai, event akan dijalankan untuk memberikan notifikasi yang dapat digunakan oleh aplikasi lain untuk melakukan pencatatan internal ataupun untuk mengolah data yang didapatkan dari notifikasi tersebut.

```
function closeCampaign(uint256 id) public {
    require(id > 0 && id <= numCampaigns, "Invalid campaign ID.");
    Campaign storage campaign = campaigns[id];
    if (block.timestamp > campaign.deadline || campaign.raised >= campaign.goal) {
        require(campaign.owner == msg.sender, "Only the campaign owner can close the campaign.");

        campaign.closed = true;
        uint256 amountRaised = campaign.raised;
        payable(campaign.owner).transfer(amountRaised);

        emit CampaignClosed(id, campaign.ownerId, amountRaised);
    }
}
```

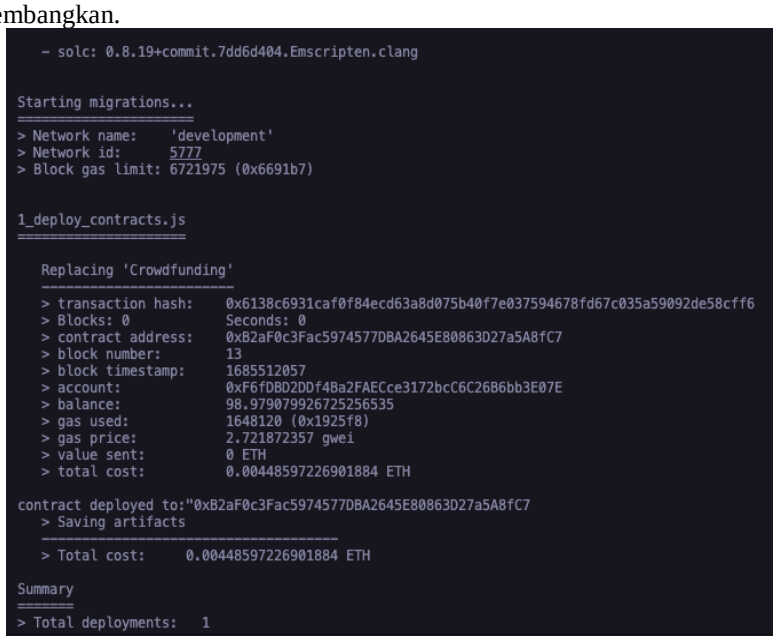
Gambar 5. Fungsi untuk Penutupan Campaign

Gambar 5 merupakan fungsi untuk melakukan proses penutupan *campaign* dari proses bisnis *fund*raising. Walaupun didalam fungsi ini terdapat perpindahan dana namun fungsi tidak mendapatkan flag payable dikarenakan perpindahan dana terjadi dari sistem. Untuk melakukan proses penutupan *campaign* ini, harus dipastikan bahwa dalam *campaign* sudah terkumpul dana sesuai dengan jumlah yang diharapkan atau lebih.

Apabila dana yang terkumpul masih belum sesuai dengan jumlah yang ingin dicapai namun batas waktu telah terlampaui, maka fungsi ini tetap dapat dijalankan. Selain itu, untuk menjalankan fungsi ini, pihak yang memiliki wewenang untuk melakukan proses pentupan *campaign* adalah pembuat *campaign.* Apabila kredensial dari pembuat dan penutup *campaign* tidak sesuai, maka dana tidak dapat dipindahkan. Setelah semua proses pengecekan selesai, maka dana yang mengendap dalam *smartcontract* akan dikirimkan secara otomatis kepada pembuat *campaign* dan notifikasi akan dikirimkan kepada aplikasi yang melakukan *subscribe* terhadap *event* fungsi ini.

3.3. Deployment

Proses *deployment* akan menggunakan bantuan dari server *develop*ment yang telah disediakan dalam internet untuk melakukan testing *smartcontract* dan transaksi untuk melihat respon dan juga perilaku dari aplikasi yang telah di dikembangkan.

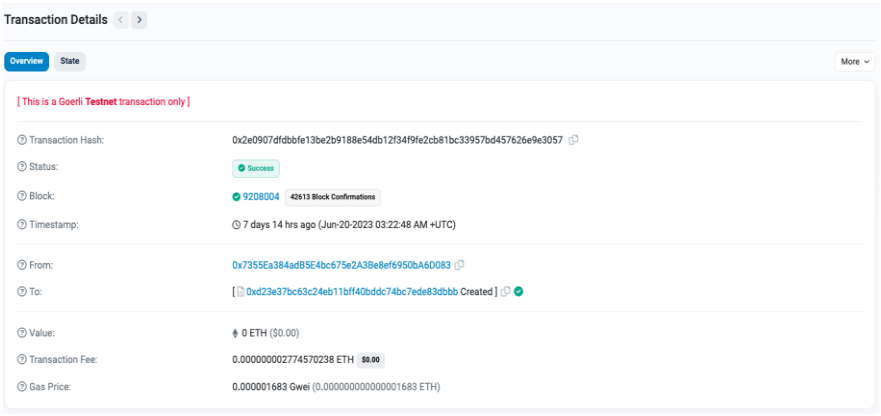


Gambar 6. Log Sukses *Deploy Smartcontract*

Dalam Gambar 6 diperlihatkan log yang akan muncul ketika proses *deploy* telah selesai dilakukan. Dalam log tersebut akan diperlihatkan beberapa detail seperti dana yang dikeluarkan untuk melakukan proses *deploy*, *address* yang menjadi identitas dari *smartcontract*, dan sebagainya. Setelah proses *deployment* telah selesai, semua proses transaksi dan kegiatan yang akan dilakukan menggunakan *smartcontract* tersebut dapat dilihat secara bebas dalam internet sehingga lebih aman dari manipulasi data dan lebih transparan.

4. HASIL DAN PEMBAHASAN

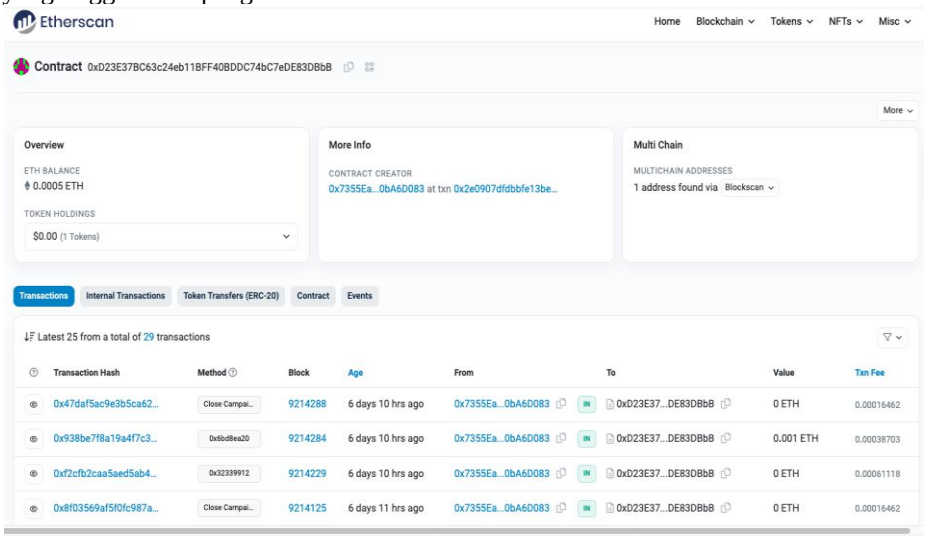
Ketika *smartcontract* berhasil di unggah kedalam jaringan *blockchain*, bukti dari transaksi tersebut dapat diakses oleh semua pengguna internet di seluruh. Gambar 7 menampilkan log yang memberikan informasi detail yang dicatat oleh jaringan *blockchain* saat *smartcontract* di unggah.



Gambar 7. Log Pembuatan *Smartcontract*

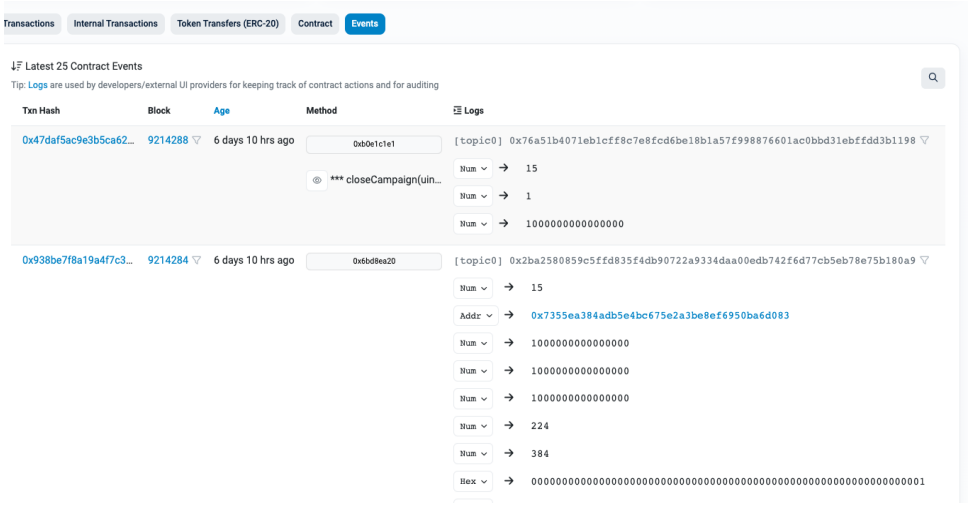
Smartcontract tersebut akan menghasilkan sebuah *address* yang akan menjadi identitas *smartcontract*. Ketika sebuah aplikasi ingin memanggil fungsi yang didefinisikan dalam *smartcontract*, aplikasi tersebut dapat mengaksesnya menggunakan *address* yang telah terintegrasi sebelumnya.

Address tersebut akan digunakan oleh *blockchain* untuk menyimpan seluruh informasi tentang transaksi aktivitas yang dilakukan menggunakan *smarcontract* yang telah di unggah. Gambar 8 menunjukakan halaman yang memberikan informasi terkait *smartcontract* berdasarkan *address* yang di cari. Dalam konteks *blockchain*, transaksi-transaksi tersebut tidak dapat dihapus dan selalu tersedia untuk dilihat secara detail meskipun data tersebut merupakan data transaksi pertama. Hal ini terjadi karena sifat *blockchain* yang mempertahankan dan mengamankan setiap transaksi yang terjadi, menciptakan jejak transparansi dan keamanan yang tinggi dalam pengelolaan data.



Gambar 8. Halaman Informasi *Smartcontract* Berdasarkan *Address*

Selain dari detail transaksi, terdapat juga detail *event* yang telah di definisikan dalam program untuk mengirimkan data ketika fungsi telah berhasil dijalankan. Detail dari *event* tersebut juga dapat dilihat dan diperiksa kembali untuk keakuratannya, sebagaimana ditunjukkan pada Gambar 9. Hal tersebut berfungsi untuk mencegah ketidakcocokan data dan memastikan integritas data yang penting dalam konteks transaksi.



Gambar 9. Log *Event Smartcontract*

5. KESIMPULAN DAN SARAN

Fitur dan pengujian dari *smartcontract* yang telah dikembangkan telah sesuai dengan semua kebutuhan proses bisnis aplikasi *fund*raising. Dengan memanfaatkan keunggulan dari *smartcontract* dan teknologi *blockchain*, semua data yang ada dalam jaringan akan aman dan tidak dapat dimanipulasi, menghilangkan kemungkinan adanya tindakan penipuan dari pihak lain.

Pemanfaatan *smartcontract* dan *blockchain* dalam *aplikasi fundraising* memberikan keamanan, keandalan dalam memproses transaksi, serta memastikan integritas data. Teknologi *blockchain* yang menggunakan prinsip desentralisasi dan mekanisme konsensus memastikan bahwa data tidak dapat diubah tanpa persetujuan mayoritas jaringan. Hal ini membantu menghilangkan risiko kecurangan atau manipulasi data oleh pihak yang tidak berwenang, memberikan kepercayaan kepada pengguna aplikasi fundraising.

Ucapan Terima Kasih

Terima kasih disampaikan kepada Tim JITEK atas dedikasinya dalam memberikan wadah kepada masyarakat untuk terus mengembangkan ilmu pengetahuan. Dukungan dan kontribusi yang diberikan oleh Tim JITEK telah membantu dalam mengembangkan literasi akademis dan memberikan banyak dampak positif pada masyarakat luas.

DAFTAR PUSTAKA

[1] Bhabendu Kumar Mohanta, Soumyashree S Panda, & Debasish Jena. (2018). *An Overview of Smart Contract and Use cases in Blockchain Technology*.

[2] Faiez Altamimi, Waqar Asif, & Muttukrishnan Rajarajan. (2020). *DADS: Decentralized (Mobile) ApplicationsDeployment System Using Blockchain*.

[3] Calvão, F. (2019*). Crypto-miners: Digital labor and the power of blockchain technology*. Economic Anthropology, 6(1), 123–134. https://doi.org/10.1002/sea2.12136

[4] Liang, W., Fan, Y., Li, K. C., Zhang, D., & Gaudiot, J. L. (2020). *Secure Data Storage and Recovery in Industrial Blockchain Network Environments*. IEEE Transactions on Industrial Informatics, 16(10), 6543–6552. https://doi.org/10.1109/TII.2020.2966069

[5] Ganindra, G., Akbar, A., & Findawati, Y. (2021). *Web-Based Programmer Performance Recording and Measurement Information System J(Case Study: PT. Bits Miliartha) Sistem Informasi Pencatatan dan Pengukuran Kinerja Programmer Berbasis Web (Studi Kasus: PT. Bits Miliartha)*. In Seminar Nasional & Call Paper Fakultas Sains dan Teknologi (Vol. 2, Issue 1).

[6] Lutfiani, N., Mariyati, M. S., Rizky, A., & Sari, A. A. (2022). Blockchain Frontier Technology (B-Front) Decentralization Of Information Using Blockchain Technology On Mobile Apps E-Journal.

Decentralization Of Information Using *Blockchain* Technology On Mobile Apps E-Journal *Blockchain* Frontier Technology (B-Front), 1(2), 90–101.

[7] Mohammed, A. H., Abdulateef, A. A., & Abdulateef, I. A. (2021, June 11). *Hyperledger, Ethereum and Blockchain Technology: A Short Overview*. HORA 2021 - 3rd International Congress on Human-Computer Interaction, Optimization and Robotic Applications, Proceedings. https://doi.org/10.1109/HORA52670.2021.9461294

[8] Nopendri, Aurachman, R., & Amrullah, M. R. (2020). *Rancangan Simulasi Penerapan Blockchain dalam Pemilihan Presiden Indonesia Blockchain Implementation Simulation for Presidential Voting in Indonesia*. Jurnal Rekayasa Sistem & Industri (JRSI), 7(1), 10. https://doi.org/10.25124/jrsi.v7i1.373

[9] Suratkar, S., Shirole, M., & Bhirud, S. (2020a, September 28). *Cryptocurrency Wallet: A Review. 4th International Conference on Computer, Communication and Signal Processing*. ICCCSP 2020. https://doi.org/10.1109/ICCCSP49186.2020.9315193

[10] Kusumajaya, R. A., Kurniawan, D., Huda, H. I., & Siswanto, E. (2020). Penerapan Management Information Systems Untuk Kelayakan Pinjaman Menggunakan Metode Fuzzy Logic Model Tsukamoto. *Kompak: Jurnal Ilmiah Komputerisasi Akuntansi*, *13*(2), 143-153.