Jurnal Riset Sains dan Teknologi
Volume 9 No. 2 September 2025, 135-147

e-ISSN: 2549-9750
p-ISSN: 2579-9118

Implementasi *Smart Contract* pada *Electronic Voting* (Studi Kasus: Pemilihan Presiden dan Wakil Presiden BEM POLNES)

Implementation of Smart Contracts in Electronic Voting (Case Study: Election of the President and Vice President POLNES Student Executive Board)

Rahmat Wahyudi[1*], Muhammad Farman Andrijasa[1], Noor Alam Hadiwijaya[1]
[1]Program Studi Teknik Informatika Multimedia, Jurusan Teknologi Informasi,
Politeknik Negeri Samarinda
Jl. Cipto Mangunkusumo, Samarinda 75131, Indonesia

*Corresponding author: w.rahmad26@gmail.com

DOI: 10.30595/jrst.v9i2.26902

Histori Artikel:

Diajukan:
11/06/2025

Diterima:
25/08/2025

Diterbitkan:
08/09/2025

ABSTRAK

Pemilihan calon Presiden dan calon Wakil Presiden Badan Eksekutif Mahasiswa (BEM) Politeknik Negeri Samarinda masih menggunakan metode konvensional berbasis kertas suara, yang rentan terhadap manipulasi, kerusakan media suara yaitu kertas dan memakan waktu dalam perhitungan suara. Penelitian ini bertujuan merancang dan menguji prototipe sistem *e-voting* berbasis *smart contract* dengan teknologi *blockchain* untuk meningkatkan efisiensi, keamanan, dan transparansi pemilihan. Menggunakan metode Waterfall, penelitian ini mencakup analisis kebutuhan, perancangan sistem dengan *flowchart* dan *use case diagram* (UML), implementasi *smart contract* menggunakan bahasa pemrograman Solidity pada jaringan ethereum lokal (Ganache), serta pengujian melalui *unit testing* menggunakan Truffle untuk *smart contract* dan pengujian *BlackBox* untuk antarmuka sistem pengguna. Hasil pengujian menunjukkan sistem mampu mengotomatisasi proses pemilihan, termasuk registrasi kandidat dan pemilih, verifikasi identitas melalui NIM, pencegahan double voting dan perhitungan suara secara otomatis. Teknologi *blockchain* memastikan imutabilitas, transparansi dan menjamin integritas data. Hasil dari penelitian ini dapat menjadi solusi alternatif yang lebih efisien, aman dan transparan, dengan meminimalisir risiko manipulasi data, penghematan biaya logistik, potensi kerusakan media suara serta percepatan proses perhitungan suara yang secara otomatis dihitung oleh *smart contract*. Secara keseluruhan, sistem ini membuktikan potensi teknologi *blockchain* dan *smart contract* sebagai alternatif modern untuk sistem pemilihan elektronik.

Kata Kunci: *E-Voting; Smart Contract; Blockchain; Waterfall; Blackbox Testing*

ABSTRACT

The election of the candidates for President and Vice President of the Samarinda State Polytechnic Student Executive Board (BEM) still relies on conventional paper-based voting, which is susceptible to manipulation, ballot damage, and time-consuming vote counting. This research seeks to develop design and test for blockchain-based e-voting system prototype utilizing smart contracts to enhance efficiency, security, and transparency. Employing the Waterfall methodology, the research includes requirement analysis, system design using flowcharts and UML use case diagrams, smart contract implementation in Solidity on a local Ethereum network (Ganache), and testing via unit testing for smart contracts using Truffle and BlackBox testing for the user interface. Results demonstrate the system's ability to automate the election process, including candidate and voter registration, identity verification via student ID, prevention of double voting, and real-time vote counting. Blockchain technology ensures immutability, transparency and guarantees data integrity. The results of this research can be an alternative solution that is more efficient, secure and

transparent, by minimizing the risk of data manipulation, saving logistics costs, potential damage to voting media and accelerating the vote counting process which is automatically calculated by smart contracts. Overall, this system proves the potential of blockchain technology and smart contracts as a modern alternative to electronic voting systems.

Keywords: *E-Voting; Smart Contract; Blockchain; Waterfall; Blackbox Testing*

1. PENDAHULUAN

Pemilihan umum (pemilu) merupakan fondasi utama dalam sistem demokrasi di Indonesia yang menjamin hak setiap warga negara untuk turut serta berpartisipasi dalam pengambilan keputusan seorang pemimpin, termasuk dalam lingkup organisasi kampus seperti pemilihan Calon Presiden dan Wakil Presiden Badan Eksekutif Mahasiswa (BEM) pada Politeknik Negeri Samarinda (POLNES).

Namun, hingga saat ini, proses pemilihan tersebut masih dijalankan dengan metode konvensional berbasis kertas suara sebagai media utama pemungutan suara. (Jamilah et al., 2021) mengemukakan bahwa pelaksanaan metode konvensional tidak hanya memakan waktu dalam proses pemungutan dan perhitungan suara, tetapi juga menimbulkan biaya logistik yang tinggi serta risiko kerusakan media suara seperti sobek atau terkena air yang menyebabkan suara dianggap tidak sah.

Untuk dapat meminimalisir kekurangan dan masalah yang terjadi pada pemilihan secara konvensional dapat menggunakan *e-voting* sebagai alternatif dari pemilihan. *E-voting* adalah sistem *voting* yang menggunakan perangkat elektronik seperti komputer atau *smartphone* untuk menghitung suara pemilih.

Dibandingkan dengan proses *voting* metode konvensional, Pemilihan elektronik sendiri memiliki sejumlah keunggulan, di mana pemilihan elektronik dapat mempercepat, mengurangi biaya dan meningkatkan tingkat akurasi dalam proses pemilihan. Namun, penggunaan *e-voting* juga menimbulkan beberapa resiko, seperti kerentanan terhadap *cyber attack* dan masalah pada integritas data.

Beberapa penelitian mengemukakan bahwa sistem pemilihan elektronik saat ini masih menghadapi kendala dalam menjaga privasi pengguna dan keamanan data, dimana sistem yang ada masih memungkinkan terjadinya kejahatan berupa *cyber attack* yang bertujuan untuk mengendalikan jalannya proses pemilihan tersebut.

Untuk meminimalkan resiko, diperlukan beberapa tindakan pengamananan yang tepat, seperti penerapan *data encryption*, autentikasi data pemilih yang kuat, serta pemantauan secara ketat terhadap sistem dengan memanfaatkan teknologi *blockchain* (Yafi *et al*., 2023). Blockchain adalah buku besar digital (digital ledger) yang menyimpan transaksi virtual secara tidak dapat diubah (immutable) dan tidak memerlukan otoritas pusat untuk mengelola data.

Setiap data pada *blockchain* disebut blok, yang dihubungkan membentuk rantai (chain) menggunakan prinsip kriptografi untuk menjamin keamanan. Setiap blok terdiri dari data, *hash* unik, dan *hash* dari blok sebelumnya, yang divalidasi melalui algoritma konsensus seperti *proof of work* atau *proof of stake* untuk memastikan integritas data tanpa keterlibatan pihak ketiga (Hasan, 2020). Salah satu platform berbasis *blockchain* adalah Ethereum.

Ethereum adalah platform blockhain yang menggunakan *smart contract* untuk dapat menciptakan suatu blok data transaksi. (Manik *et al.,* 2019). Smart contract adalah kontrak pintar yang dijalankan secara otomatis berdasarkan ketentuan dalam kode program, memastikan transaksi yang transparan, aman, dan tidak dapat diubah (Lim *et al*., 2024).

Penelitian ini bertujuan untuk merancang, membangun dan menguji prototipe sistem *e-voting* berbasis *smart contract* yang dijalankan secara lokal melalui website, menggunakan teknologi *blockchain* sebagai platform utama. Ganache digunakan sebagai jaringan simulasi lokal, Smart contract ditulis menggunakan bahasa pemrograman *Solidity* dan Truffle Framework untuk pengujian serta implementasi *smart contract* dengan MetaMask sebagai jembatan interaksi antara pengguna dan sistem *blockchain*.

Beberapa penelitian sebelumnya, seperti (Hu *et al*., 2019), menunjukkan bahwa *blockchain* dapat menjamin integritas data dalam sistem pemilu, sedangkan (Pratama & Kurniadi, 2021) membuktikan bahwa *smart contract* mampu mengotomatisasi proses pemilihan dan meminimalisir manipulasi. (Gupta *et al*., 2023) juga menyoroti pentingnya keamanan dan privasi pemilih dalam sistem pemilu terdesentralisasi.

Namun, penelitian-penelitian tersebut masih bersifat umum dan belum disesuaikan

dengan konteks pemilihan organisasi mahasiswa. Dengan demikian, penelitian dapat menjadi tawaran berupa solusi dalam bentuk prototipe sistem *e-voting* yang disesuaikan dengan kebutuhan pemilihan calon Presiden dan Wakil Presiden Badan Eksekutif Mahasiswa (BEM) pada Politeknik Negeri Samarinda.

Sistem ini dirancang untuk menunjukkan penerapan *smart contract*, sekaligus memberikan pemahaman teknis tentang integrasi *blockchain* dalam pengembangan sistem pemilihan elektronik *(e-voting).*

2. METODE PENELITIAN

Metode penelitian ini menggunakan pendekatan metode *waterfall*. (Kurniawan, *et al.,* 2020) mengemukakan bahwa Metode *waterfall* adalah tahapan yang menggambarkan pendekatan secara sistematis dan juga berurutan *(step by step)* pada sebuah pengembangan perangkat lunak.

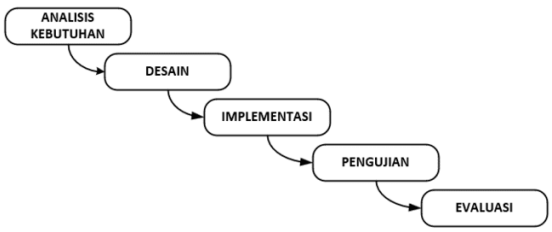
2.1 Pengumpulan Data

Pengumpulan data digunakan untuk mendapatkan informasi yang diperlukan melalui teknik-teknik yang akan diterapkan untuk memperoleh data relevan yang meliputi:

a. Observasi dilakukan untuk pengamatan secara langsung terhadap proses pemilihan Presiden dan Wakil Presiden BEM POLNES.
b. Pengumpulan data primer Mahasiswa untuk keperluan pengujian sistem, dikumpulkan data primer berupa sampel data mahasiswa aktif POLNES dari BEM.
c. Pencarian informasi data sekunder melalui studi literatur dari sumber-sumber referensi yang relevan dengan topik penelitian.

2.2 Tahapan Penelitian

Tahapan penelitian dapat dilihat pada **Gambar 1** dibawah ini.



Gambar 1. Tahapan Penelitian

a. Analisis kebutuhan

Dilakukan untuk mengetahui dan mengumpulkan semua kebutuhan yang diperlukan agar sesuai dengan kebutuhan pengguna dan memastikan bahwa sistem yang akan dibangun dapat berfungsi dengan baik.

b. Desain

Dilakukan perancangan desain arsitektur sistem yang meliputi pembuatan *flowchart,* UML *(use case diagram)* dan antarmuka pengguna.

c. Implementasi

Dilakukan penerapan *smart contract* ke dalam sistem untuk merealisasikan desain arsitektur yang telah dirancang sebelumnya.

d. Pengujian

Dilakukan pengujian menyeluruh terhadap sistem yang telah dibuat. Pengujian dibagi menjadi dua bagian utama yang meliputi pengujian *smart contract* menggunakan Truffle Framework dan pengujian sistem antarmuka pengguna dengan *BlackBox Testing*, untuk memastikan bahwa semua fungsionalitas sistem berhasil diterapkan.

e. Evaluasi

Dilakukan untuk menilai apakah sistem sudah memenuhi kebutuhan dan berfungsi sesuai dengan yang di ekspektasikan/diharapkan.

3. HASIL DAN PEMBAHASAN

3.1 Analisis Kebutuhan

Sistem pemilihan elektronik *(e-voting)* berbasis *smart contract* dirancang untuk memenuhi:

a. Kebutuhan Fungsional

Registrasi kandidat dan pemilih, verifikasi identitas pemilih melalui NIM, pencegahan *double voting*, penghitungan suara otomatis dan pencatatan laporan.

b. Kebutuhan Non-Fungsional

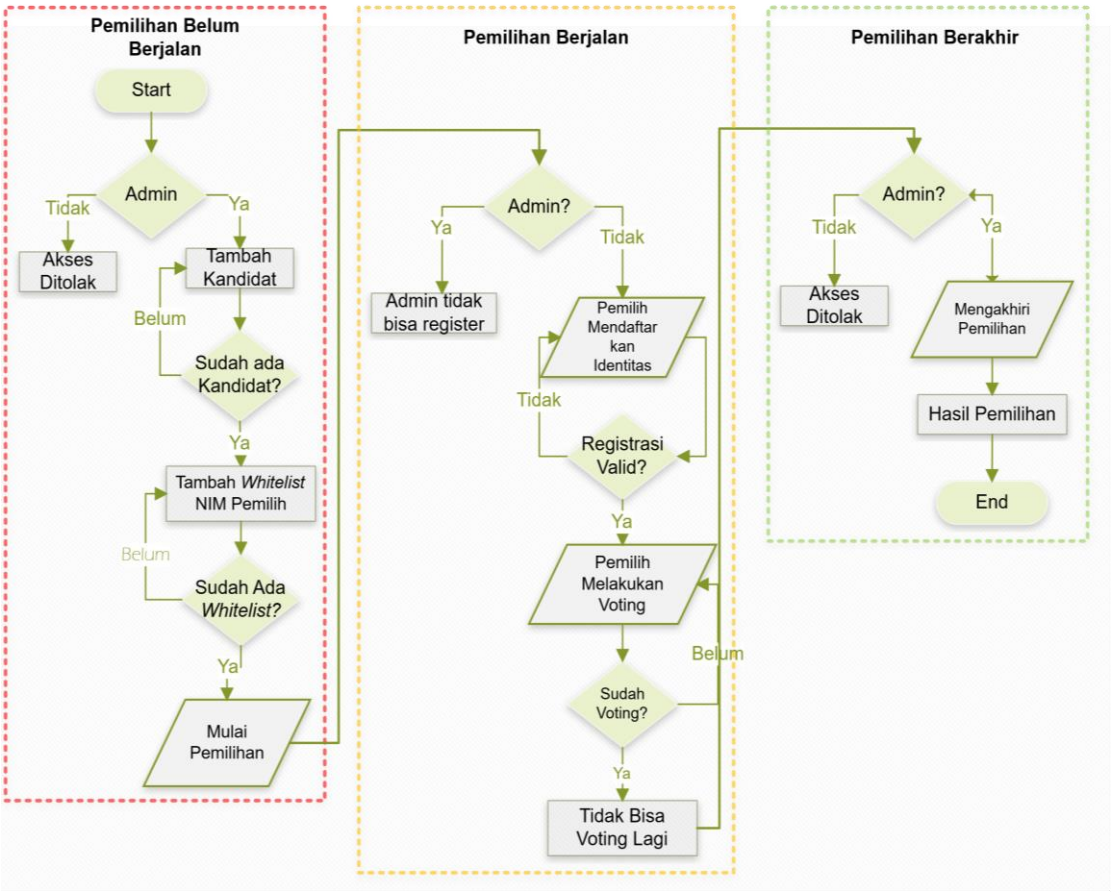
Smart contract dapat mengotomatisasi dan menjamin proses pemilihan sesuai aturan yang telah ditetapkan.

c. Kebutuhan Pengguna.

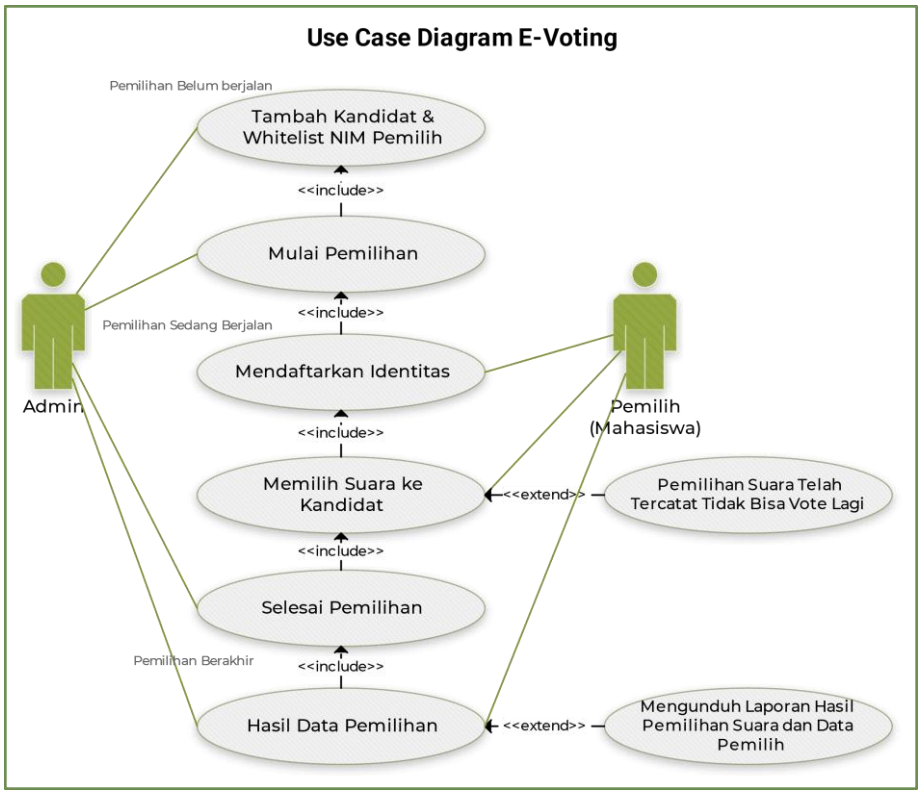
Admin: Manajemen kandidat, memulai dan mengakhiri pemilihan, dan memantau hasil. Pemilih (Mahasiswa): Memberikan suara hanya sekali dengan verifikasi Nomor Induk Mahasiswa.

3.2 Desain *Flowchart*

Flowchart merupakan representasi visual secara grafik dari langkah-langkah dan urut-urutan prosedur dari suatu program (Zalukhu *et al.,* 2023). Penerapan *Flowchart* pada penelitian ini meliputi tiga fase, yaitu Pemilihan Belum Berjalan, Pemilihan Berjalan dan Pemilihan Berakhir. Berikut *flowchart* sistem dapat terlihat pada **Gambar 2** dibawah ini.



Gambar 2. *Flowchart Sistem*



Gambar 3. *Use Case Diagram*

3.3 Desain *Use Case Diagram* (UML)

UML *(Unified Modeling Language)* adalah sebuah standar Bahasa yang digunakan untuk merancang dan menganalisis serta menggambarkan arsitektur program dalam *object oriented programming* (Kurniawan *et al.,* 2020).

Use case diagram adalah gambaran grafis yang menggambarkan sebagian atau seluruh use case, aktor, serta interaksi dalam suatu sistem. Diagram ini tidak memberikan penjelasan rinci tentang cara kerja use case, melainkan hanya menyajikan gambaran umum mengenai hubungan antara use case, aktor dan sistem (Pratama et al., 2019). Penerapan use case diagram pada penelitian ini dapat terlihat pada **Gambar 3**.

Penulisan *Smart Contract*

Smart contract ditulis menggunakan bahasa pemrograman Solidity. Solidity adalah bahasa pemrograman berorientasi objek yang digunakan pada Ethereum Virtual Machine (EVM) dan disimpan dalam ekstensi (.sol). Solidity *code* dikompilasi menggunakan Solidity compiler atau "solc," menghasilkan kode byte *(bytecode)* yaitu sekumpulan fungsi yang telah dienkode (Wirayudha, 2024).

Penulisan *smart contract* pada penelitian ini membuat 6 fungsi yang dapat dilihat oleh **Gambar 4**, **Gambar 5**, **Gambar 6**, **Gambar 7**, **Gambar 8**, **Gambar 9** berikut.

a. Fungsi Tambah Kandidat

```
function addCandidate(
        string memory _Cname, uint _CNIM, string memory _Cjurusan, string memory _Cprodi,
        string memory _Wname, uint _WNIM, string memory _Wjurusan, string memory _Wprodi,
        string memory _photoPath
    ) public onlyAdmin electionNotOngoing nonReentrant {
        require(bytes(_Cname).length > 0, "Nama Calon Presiden Kandidat di butuhkan.");
        require(bytes(_Wname).length > 0, "Nama Calon Wakil Presiden Kandidat di butuhkan.");
        require(_CNIM > 0, "NIM Calon Presiden Kandidat di butuhkan.");
        require(_WNIM > 0, "NIM Calon Wakil Presiden Kandidat di butuhkan.");
        require(bytes(_Wjurusan).length > 0, "Jurusan Calon Wakil di butuhkan.");
        require(bytes(_Wprodi).length > 0, "Prodi Calon Wakil Presiden di butuhkan.");
        require(_CNIM != _WNIM, "NIM calon Presiden dan NIM Wakil Tidak Bisa Sama.");
        require(candidatesByNIM[_CNIM] == 0, "NIM Calon Presiden Telah Ada.");
        require(candidatesByWNIM[_WNIM] == 0, "NIM Wakil Calon Presiden Telah ada." );
        require(candidatesByWNIM[_CNIM] == 0, "NIM Calon Presiden telah ada untuk Calon Wakil Presiden.");
        require(candidatesByNIM[_WNIM] == 0, "NIM Wakil Calon Presiden telah ada untuk Calon Presiden.");

        candidatesCount++;
        bytes32 candidateId = keccak256(
            abi.encodePacked(_CNIM, _WNIM, msg.sender)
        );

        candidates[candidatesCount] = ElectionStructs.Candidate(
            candidateId, _Cname, _CNIM, _Cjurusan, _Cprodi, _Wname,
            _WNIM, _Wjurusan, _Wprodi, _photoPath, 0
        );
        candidateIndexById[candidateId] = candidatesCount;
        candidatesByNIM[_CNIM] = candidatesCount;
        candidatesByWNIM[_WNIM] = candidatesCount;

        emit ElectionEvents.AddCandidate(candidateId, _CNIM, _WNIM, _photoPath);
    }
```

Gambar 4. Fungsi Tambah Kandidat

b. Fungsi set whitelist daftar NIM voter

```
function setWhitelistNIM(
        bytes32 _merkleRoot
    ) public onlyAdmin electionNotOngoing nonReentrant {
        require(
            whitelistNIM == bytes32(0),
            "Whitelist Pemilih telah diatur dan tidak dapat diubah"
        );
        whitelistNIM = _merkleRoot;
        emit ElectionEvents.setWhitelistNIM(_merkleRoot);
    }
```

Gambar 5. Fungsi set whitelist daftar nim

c. Fungsi mulai proses pemilihan

```
function startElection() public onlyAdmin electionNotOngoing nonReentrant {
    require(candidatesCount > 0, "Pemilihan Tidak Bisa dimulai jika Tidak ada Kandidat");
    require( whitelistMerkleRoot != bytes32(0), "Belum Menambahkan Whitelist");
    electionStarted = true;
    electionEnded = false;

    emit ElectionEvents.ElectionStarted();
}
```

Gambar 6. Fungsi mulai proses pemilihan

d. Fungsi Pengisian Informasi Identitas Pemilih (Mahasiswa)

```
function addUser(
        string memory _name, uint _NIM,
        string memory _jurusan, string memory _prodi,
        bytes32[] calldata _merkleProof
    ) public notAdmin electionOngoing nonReentrant {
        bytes32 leaf = keccak256(abi.encodePacked(_NIM));
        require(
            MerkleProof.verify(_merkleProof, whitelistNIM, leaf),
            "NIM tidak masuk Whitelist"
        );
        require(bytes(_name).length > 0, "Nama dibutuhkan");
        require(_NIM > 0, "NIM dibutuhkan");
        require(userByNIM[_NIM] == 0, "Pemilih dengan NIM ini telah ada");
        require(userByAddress[msg.sender] == 0, "Address telah terdaftar");
        usersCount++;
        users[usersCount] = ElectionStructs.User(
            _name, _NIM, _jurusan,
            _prodi, msg.sender, 0
        );

        userByNIM[_NIM] = usersCount;
        userByAddress[msg.sender] = usersCount;

        emit ElectionEvents.AddUser(
            _NIM, _name, _jurusan, _prodi, msg.sender, 0
        );
    }
```

Gambar 7. Fungsi tambah pemiliih *(voter)*

e. Fungsi vote kandidat

```
function vote(bytes32 _candidateId)
    public notAdmin electionOngoing nonReentrant {
        require(!hasVoted[msg.sender], "Pemilih telah Melakukan Voting");
        uint userIndex = userByAddress[msg.sender];
        require(userIndex != 0, "Pemilih Belum Mendaftar atau Register");
        ElectionStructs.User storage user = users[userIndex];
        require(
            user.add == msg.sender,
            "Ketidakcocokan address untuk pengguna terdaftar"
        );
        uint candidateIndex = candidateIndexById[_candidateId];
        require(candidateIndex != 0, "Kandidat Tidak Ditemukan");
        hasVoted[msg.sender] = true;
        candidates[candidateIndex].voteCount++;
        totalVotes++;
        user.voteTimestamp = block.timestamp;

        emit ElectionEvents.CastVote(_candidateId, msg.sender, block.timestamp);
    }
```

Gambar 8. Fungsi *vote* kandidat

f. Fungsi mengakhiri proses pemilihan

```
function endElection() public onlyAdmin electionOngoing {
    electionEnded = true;
    emit ElectionEvents.ElectionEnded();
}
```

Gambar 9. Fungsi mengakhiri proses pemilihan

```
D:\evoting>truffle compile

Compiling your contracts...
===========================
> Compiling .\contracts\Election.sol
> Compiling .\contracts\Election.sol
> Compiling .\contracts\libraries\ElectionEvents.sol
> Compiling .\contracts\libraries\ElectionEvents.sol
> Compiling .\contracts\libraries\ElectionStructs.sol
> Artifacts written to D:\evoting\build\contracts
> Compiled successfully using:
   - solc: 0.8.20+commit.a1b79de6.Emscripten.clang
```

Gambar 10. Kompilasi *smart contract*

Setiap fungsi tersebut dirancang untuk memastikan bahwa tahapan-tahapan pemilihan dari pra, proses dan pasca berjalan sesuai alur yang telah ditentukan secara otomatis melalui *smart contract*.

3.4 Penerapan *Smart Contract* ke *Blockchain* menggunakan Truffle Framework

Truffle Framework digunakan sebagai *tool* untuk mempermudah proses kompilasi, migrasi dan deployment ke jaringan ethereum pribadi lokal (Gupta *et al.,* 2020). Ganache berfungsi sebagai jaringan *blockchain* lokal yang mensimulasikan *blockchain* ethereum sehingga pengembang dapat menguji *smart contract* mereka di lingkungan yang aman sebelum meluncurkannya ke jaringan utama ethereum (Yogiyanti & Suartana, 2024).

Penerapan *smart contract* pada penelitian ini memuat 2 tahapan, yaitu kompilasi *(compile) smart contract* dan migrasi & deploy *smart contract*.

a. Kompilasi Smart Contract

Smart contract yang ditulis dalam *Solidity* dikompilasi menggunakan *truffle* untuk memastikan tidak ada kesalahan sintaksis dan memastikan *smart contract* siap digunakan. Kompilasi *smart contract* dapat dilihat pada **Gambar 1**.

b. Migrasi dan deployment smart contract

Pada migrasi, bytecode disimpan ke dalam blok di *blockchain* dan setelah deployment berhasil, alamat kontrak (contract address) yang unik dihasilkan untuk interaksi dengan *smart contract*.

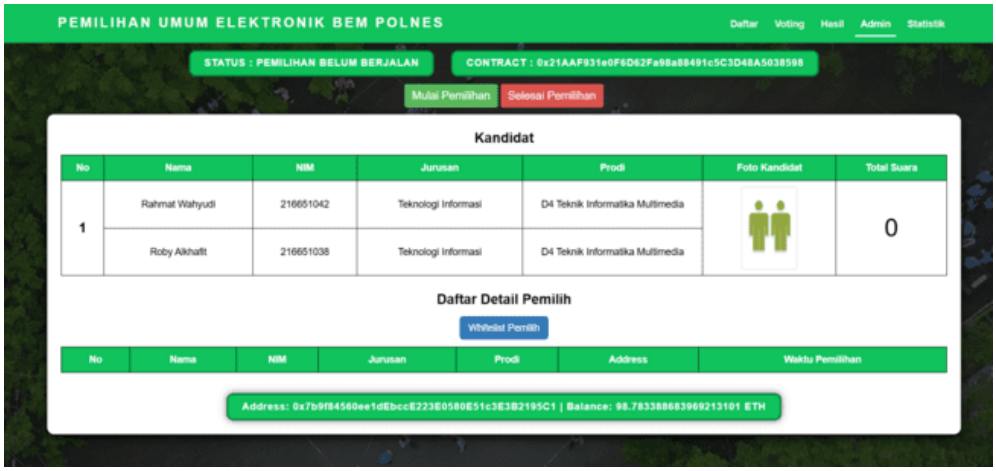
Migrasi dan deployment *smart contract* dapat dilihat pada **Gambar 11**.

Gambar 11. Migrasi dan deployment *smart contract*

3.5 Penerapan Antarmuka Sistem

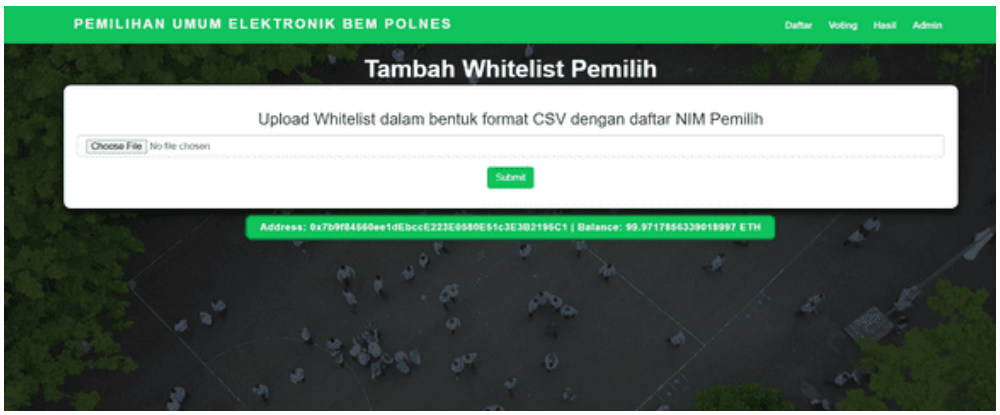
Pada penerapan antarmuka sistem pada penelitian ini dibangun 6 tampilan utama yang dapat dilihat oleh **Gambar 12**, **Gambar 13**, **Gambar 14**, **Gambar 15**, **Gambar 16**, **Gambar 17**, **Gambar 18**.

a. Halaman Administrator



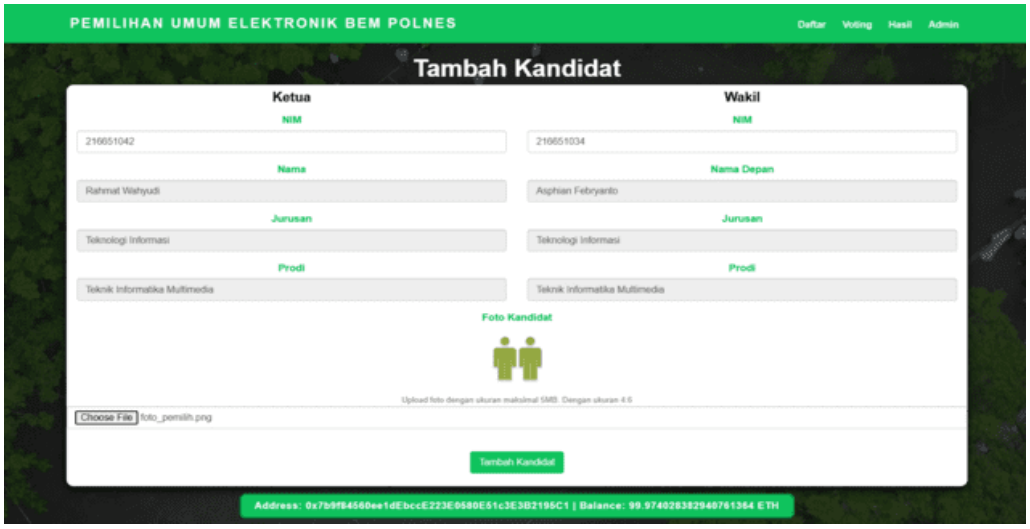
Gambar 12. Halaman administrator

b. Halaman Tambah Daftar Whitelist NIM Pemilih (voters)



Gambar 13. Halaman Tambah Daftar Whitelist Pemilih

c. Halaman Tambah Kandidat

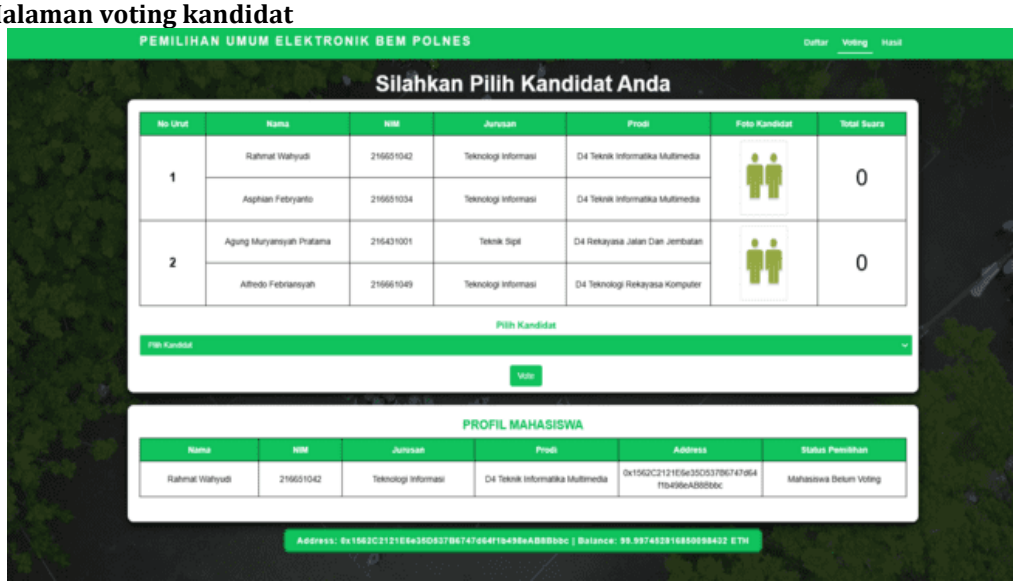


Gambar 14. Halaman Tambah Kandidat

d. Halaman Pengisian Informasi Identitas Pemilih (Mahasiswa)

Gambar 15. Halaman Pengisian Informasi Identitas Pemilih (Mahasiswa)

e. Halaman voting kandidat

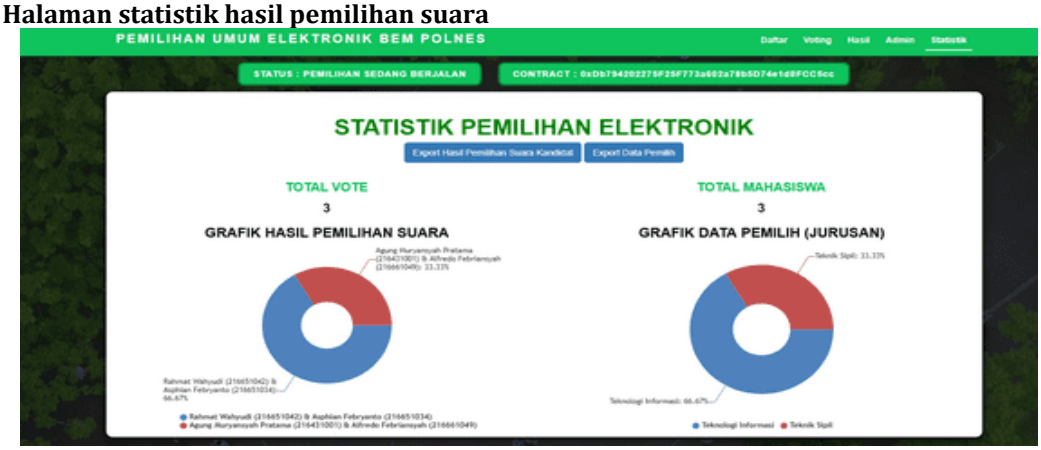


Gambar 16. Halaman pemilih *vote* kandidat

f. Halaman hasil pemilihan suara

Gambar 17. Halaman hasil pemilihan suara

g. Halaman statistik hasil pemilihan suara



Gambar 18. Halaman statistik pemilihan

Gambar 19. Hasil *unit testing smart contract*

3.6 Pengujian

a. *Unit Testing* Fungsional dan Non-Fungsional *Smart Contract*

Unit Testing dilakukan untuk memastikan *smart contract* berfungsi secara fungsional pada *smart contract* yang telah dibuat, serta aspek non-fungsional untuk menguji keamanan *smart contract*. Pengujian dilakukan menggunakan Truffle framework.

Pada penelitian ini, hasil dari unit testing yang telah dibuat dari berbagai skenario fungsional dan non-fungsional dapat terlihat pada **Gambar 19**.

b. *BlackBox Testing* Antarmuka *Sistem*

Pengujian ini bertujuan untuk memastikan bahwa setiap proses dalam perangkat lunak berfungsi sesuai dengan yang diharapkan. Penguji akan menguji berbagai kondisi masukan dan memeriksa apakah fungsi-fungsi spesifik dalam sistem berjalan dengan benar. Pengujian ini bertujuan untuk menemukan kesalahan atau masalah, lalu memperbaikinya sehingga sistem bisa dinyatakan siap dan aman digunakan (Wijaya & Astuti, 2021). Pada penelitian ini, hasil dari BlackBox testing yang telah dibuat dapat terlihat pada **Tabel 1**, **Tabel 2**, **Tabel 3**, **Tabel 4**, **Tabel 5**, **Tabel 6**, **Tabel 7**.

Tabel 1. *BlackBox Testing* Halaman Administrator

No	Fungsi	Input	Output	Hasil
1	Status Pemilihan	-	Menampilkan status pemilihan	Berhasil
2	Data Kandidat & Data Pemilih	-	Menampilkan Data Kandidat & Data Pemilih	Berhasil
3	Mulai Pemillihan	Interaksi Metamask Pengguna	Pemilihan Sedang Berjalan	Berhasil

Tabel 2. *BlackBox Testing* Halaman Tambah Kandidat

No	Fungsi	Input	Output	Hasil
1	Form Tambah Kandidat	Isi Data Kandidat	Data Kandidat Calon Presiden dan Calon Wakil Presiden	Berhasil
2	Tambah Kandidat	Interaksi Metamask Pengguna	Kandidat Berhasil di Tambah	Berhasil

Tabel 3. BlackBox Testing Halaman Tambah Daftar whitelist NIM Pemilih *(voters)*

No	Fungsi	Input	Output	Hasil
1	Form Input File (.csv)	Daftar Whitelist NIM Semua Pemilih	-	Berhasil
2	Tambah Whitelist Daftar NIM Pemilih	Interaksi Metamask Pengguna	Daftar Semua Whitelist Daftar NIM Pemilih berhasil di Tambah	Berhasil

Tabel 4. BlackBox Testing Halaman Pengisian Informasi Identitas Pemilih (Mahasiswa)

No	Fungsi	Input	Output	Hasil
1	Form Tambah Pemilih	Isi NIM Pemilih	Menampilkan Data Pemilih	Berhasil
2	Tambah Pemilih	Interaksi Metamask Pengguna	Pemilih Berhasil di Tambah	Berhasil

Tabel 5. BlackBox Testing Halaman pemilih voting kandidat

No	Fungsi	Input	Output	Hasil
1	Data Kandidat dan Data Pemilih	-	Menampilkan Data Kandidat dan Data Pemilih	Berhasil
2	Form Input Select Box Kandidat	Kandidat yang Dipilih	-	Berhasil
3	Pilih Kandidat	Interaksi Metamask Pengguna	Kandidat Berhasil Dipilih	Berhasil

Tabel 6. BlackBox Testing Halaman hasil pemilihan suara

No	Fungsi	Input	Output	Hasil
1	Data Kandidat dan total Suara	-	Menampilkan Data Kandidat dan Total Suara yang Diperoleh	Berhasil

Tabel 7. BlackBox Testing Halaman statistik hasil pemilihan suara

No	Fungsi	Input	Output	Hasil
1	Data Kandidat dan Data Pemilih secara Detail	-	Menampilkan Data Kandidat dan Data Pemilih secara Detail	Berhasil
2	Grafik Data Pemilihan Suara, Kandidat dan Pemilih	-	Menampilkan Grafik Data Pemilihan Suara, Kandidat dan Pemilih	Berhasil
3	Unduh Laporan Suara pada Kandidat (.pdf)	-	Laporan Berhasil di Unduh (.pdf)	Berhasil
4	Unduh Laporan Daftar Pemilih (.pdf)	-	Laporan Berhasil di Unduh (.pdf)	Berhasil

3.7 Evaluasi

Berdasarkan hasil pengujian *unit testing*, semua fungsi utama *smart contract* berjalan sesuai dengan aturan yang ditetapkan. Pengujian dilakukan pada bagian-bagian penting seperti memulai proses pemilihan, pendaftaran pemilih, pendaftaran *whitelist* NIM pemilih, proses *voting*, mengakhiri proses pemilihan, pengecekan hasil suara dan semua aspek keamanan yang diperlukan. Semua bagian tersebut berhasil melewati pengujian tanpa terjadi *error* atau kesalahan proses.

Selain itu, hasil dari pengujian *BlackBox* pada antarmuka sistem untuk pengguna. menunjukkan bahwa sistem dapat merespons input dari pengguna dengan benar. Fitur-fitur seperti pengisian identitas pemilih, proses pemilihan kandidat oleh pemilih dan tampilan hasil pemilihan suara dapat digunakan dengan baik tanpa mengalami kendala. Hal ini menunjukkan bahwa sistem sudah cukup baik dalam mendukung proses *e-voting* dari sisi pengguna.

4. SIMPULAN

Sistem berhasil dirancang, diimplementasikan dan diuji dengan baik. Sistem mampu mengotomatisasi proses pemilihan melalui *smart contract* yang dikembangkan menggunakan Solidity pada jaringan *blockchain* pribadi (Ganache) yang dijalankan secara lokal melalui website. Dari sisi *smart contract* sistem berhasil memenuhi kebutuhan fungsional dan non-fungsional, seperti registrasi kandidat dan pemilih, verifikasi identitas pemilih melalui NIM, pencegahan *double voting*, serta perhitungan suara secara otomatis.

Dari sisi *blockchain*, menjamin keamanan melalui imutabilitas, integritas data dan transparansi. Pengujian unit testing pada *smart contract* dan *BlackBox testing* pada antarmuka sistem menunjukkan bahwa semua fungsi berjalan sesuai skenario dan aturan yang telah ditetapkan.

Dibandingkan metode konvensional berbasis kertas, prototipe ini dapat menjadi solusi yang lebih efisien, aman dan transparan, dengan meminimalisir risiko manipulasi data, penghematan biaya logistik, potensi kerusakan media suara serta percepatan proses perhitungan suara yang secara otomatis dihitung oleh *smart contract*. Secara keseluruhan, sistem ini membuktikan potensi teknologi *blockchain* dan *smart contract* sebagai alternatif modern untuk sistem pemilihan elektronik.

Untuk pengembangan lebih lanjut, penelitian selanjutnya dapat mengimplementasikan jaringan *blockchain* publik layer 2 seperti Optimism, Base, atau ZKSync untuk meningkatkan skalabilitas dan efisiensi sumber daya, sehingga sistem dapat diuji pada lingkungan yang lebih besar dan realistis, serta mengintegrasikan WalletConnect QR untuk mempermudah autentikasi pemilih melalui dompet digital Web3 seperti MetaMask atau TrustWallet guna meningkatkan keamanan dan kenyamanan pengguna, sambil menerapkan teknologi Zero-Knowledge Proof (ZK Proof) untuk memastikan privasi pemilih dengan validasi identitas tanpa mengungkapkan data sensitif, sehingga menjaga anonimitas sekaligus keabsahan suara.

REFERENSI

Gupta, R., Jha, B., Shukla, A. K., Raj, A., & Sultana, S. (2020). *Secure and Decentralized Smart Elections. IOSR Jurnal of Computer Engineering,* 22(4), 52–57. https://doi.org/10.9790/0661-2204015257

Hasan, N. (2020). *Blockchain Technology and its Application in Libraries. Library* Herald, 58(4), 118–125. https://doi.org/10.5958/0976-2469.2020.00036.6

Jamilah, S., Faruq, H. A. al, & Wijaya, G. (2021). Perancangan Sistem E-Voting Berbasis Web pada Pemilihan BEM (Badan Eksekutif Mahasiswa) Fakultas Teknik Universitas Muhammadiyah Jember. Jurnal *Smart* Teknologi, 1(1), 100–110. http://jurnal.unmuhjember.ac.id/index.php/JST

Kurniawan, H., Apriliah, W., Kurniawan, I., & Firmansyah, D. (2020). Penerapan Metode *Waterfall* Dalam Perancangan Sistem Informasi Penggajian Pada SMK Bina Karya Karawang. Jurnal Interkom: Jurnal Publikasi Ilmiah Bidang Teknologi Informasi Dan Komunikasi, 14(4), 159–169. https://doi.org/10.35969/interkom.v14i4.58

Wijaya, Y. D., & Astuti, M. W. (2021). Pengujian Blackbox Sistem Informasi Penilaian Kinerja Karyawan Pt Inka (Persero) Berbasis Equivalence Partitions. Jurnal Digital Teknologi Informasi, 4(1), 22–26. https://doi.org/10.32502/digital.v4i1

Wingga Pratama, Y., & Kurniadi, D. (2021). Implementasi Blockchain dalam Aplikasi Pemilu. *Vocational* Teknik Elektronika Dan Informatika, 9(3), 123–130. http://ejournal.unp.ac.id/index.php/voteknika/ https://doi.org/10.24036/voteteknika.v9i3.113787

Yafi, A., Arhandi, P. P., Firdaus, V. A. H., Ismail, A., & Suarjuna Batubulan, K. (2023). Sistem Keamanan *E-Voting* Menggunakan Arsitektur Publik Blockchain Ethereum. KLIK: Kajian Ilmiah Informatika Dan Komputer, 4(3), 1313–1322. https://doi.org/10.30865/klik.v4i3.1423

Yogiyanti, E., & Suartana, I. M. (2024). Penerapan Teknologi Blockchain pada Sistem Laporan Keuangan Aplikasi *Point of Sale*. Journal of Informatics and Computer Science, 6(1), 96–104. https://doi.org/10.26740/jinacs.v6n01.p96-104

Zalukhu, A., Purba, S., & Darma, D. (2023). Perangkat Lunak Aplikasi Pembelajaran Flowchart. Jurnal Teknologi Informasi Dan Industri, 4(1), 61–70.