JSAI : Journal Scientific and Applied Informatics
Vol. 09, No. 3, *October* 2026, *pages*. 416~425
E-ISSN: 2614-3054; P-ISSN: 2614-3062, *accredited by Kemenristekdikti*, Sinta 4
DOI: 10.36085

Verifikasi Integritas Kuitansi Digital Sewa Kos Berbasis *Hash Anchoring Blockchain Ethereum* dan *Payment Gateway Midtrans*

[1]Seno Marky Pallawagau, [2]Donny Avianto
[1,2]Universitas Teknologi Yogyakarta, Indonesia
[1]*senopallawagau9@gmail.com*; [2]*donny@uty.ac.id*;

Article Info

Article history:

Received, 2026-10-14
Revised, 2026-10-27
Accepted, 2026-11-04

Kata Kunci:

Smart Contract
Deteksi Manipulasi Data
Arsitektur Hibrida
Jaringan Sepolia
Redundansi Relasional

ABSTRAK

Penelitian ini mengembangkan mekanisme verifikasi integritas kuitansi digital sewa kos dengan mengintegrasikan *payment gateway Midtrans*, hash SHA-256, dan smart contract pada jaringan uji *Ethereum Sepolia*. Data operasional disimpan pada basis data MySQL, sedangkan hash dari 17 atribut kuitansi dicatat pada *blockchain*. Sistem dilengkapi *Diagnostic Tamper Engine* yang memanfaatkan relasi antartabel untuk membantu mengidentifikasi atribut yang berubah. Evaluasi dilakukan melalui enam transaksi pencatatan, empat skenario injeksi manipulasi, satu skenario kontrol, dan 32 skenario pengujian fungsional. Hasil menunjukkan bahwa seluruh transaksi berhasil dicatat (6/6; 100%) dengan konsumsi 576.856 gas per transaksi dan latensi konfirmasi rata-rata 14,00 detik. Latensi verifikasi baca rata-rata sebesar 277,0 ms. Seluruh empat skenario manipulasi berhasil dideteksi dan didiagnosis (4/4; tingkat keberhasilan deteksi 100%), sedangkan skenario kontrol tetap terverifikasi. Seluruh pengujian fungsional juga berhasil dilaksanakan (32/32; 100%). Hasil ini menunjukkan bahwa mekanisme yang dikembangkan dapat memverifikasi integritas kuitansi dan mengenali perubahan pada skenario terkontrol. Nilai 100% tersebut terbatas pada skenario yang diuji dan belum mewakili akurasi pada seluruh kemungkinan manipulasi.

Keywords:

Smart Contract
Data Tampering Detection
Hybrid Architecture
Sepolia *Network*
Relational Redundancy

ABSTRACT

This study develops a digital boarding-house receipt integrity verification mechanism by integrating the Midtrans payment gateway, SHA-256 hashing, and a smart contract on the Ethereum Sepolia test network. Operational data are stored in a MySQL database, while hashes of 17 receipt attributes are anchored on the blockchain. The system includes a Diagnostic Tamper Engine that uses inter-table relationships to help identify modified attributes. Evaluation consisted of six hash-recording transactions, four database-tampering scenarios, one control scenario, and 32 functional test scenarios. All recording transactions succeeded (6/6; 100%), consuming 576,856 gas per transaction with an average confirmation latency of 14.00 seconds. The average read-verification latency was 277.0 ms. All four tampering scenarios were detected and diagnosed (4/4; a 100% detection success rate), while the control scenario remained verified. All functional test scenarios also passed (32/32; 100%). These results indicate that the proposed mechanism can verify receipt integrity and identify changes under the controlled scenarios tested. The reported 100% rate is limited to these test cases and does not represent accuracy across all possible forms of tampering.

This is an open access article under the CC BY-SAlicense.

Penulis Korespondensi:

Seno Marky Pallawagau,
Program Studi Informatika,
Universitas Teknologi Yogyakarta,
Email: senopallawagau9@gmail.com

1. PENDAHULUAN

Hunian kos menjadi salah satu pilihan tempat tinggal sementara bagi mahasiswa, pekerja, dan masyarakat yang beraktivitas jauh dari tempat tinggal asalnya. Pengelolaan hunian kos melibatkan pencatatan penghuni, kamar, tarif, masa sewa, tagihan berkala, pembayaran, dan penerbitan kuitansi. Pemanfaatan sistem informasi dan

layanan pembayaran elektronik dapat membantu mengintegrasikan proses administrasi tersebut serta mengurangi ketergantungan pada pencatatan dan konfirmasi pembayaran secara manual [1], [2], [3]. Meskipun demikian, keberhasilan transaksi elektronik tidak secara otomatis menjamin bahwa data pembayaran dan kuitansi yang tersimpan akan tetap utuh setelah transaksi selesai.

Kuitansi pembayaran kos umumnya dibagikan dalam bentuk berkas PDF, gambar, atau tangkapan layar. Format digital memudahkan penyimpanan dan distribusi, tetapi tampilan dokumen tidak dengan sendirinya membuktikan bahwa nominal, tanggal pembayaran, tarif, dan periode sewa masih sama dengan data ketika kuitansi diterbitkan. Isi dokumen digital dapat disunting menggunakan perangkat lunak pengolah dokumen atau gambar, sehingga pihak penerima memerlukan cara untuk membandingkan dokumen yang diterima dengan bukti integritas yang dibuat pada saat penerbitan. Permasalahan pemalsuan dan perubahan dokumen digital telah dikaji melalui pendekatan deteksi pemalsuan dokumen akademik [4] serta mekanisme pertukaran dokumen PDF yang menyimpan hash dan metadata pada blockchain [5].

Risiko perubahan juga tidak terbatas pada berkas kuitansi, tetapi dapat terjadi pada data transaksi dalam basis data operasional. Basis data relasional tetap dibutuhkan untuk mengelola hubungan antara penghuni, kamar, tagihan, pembayaran, dan masa sewa. Namun, apabila nilai pada basis data operasional menjadi satu-satunya sumber pemeriksaan, perubahan oleh pihak yang memiliki akses memadai dapat menyulitkan proses pembuktian terhadap kondisi data sebelumnya. Pendekatan kriptografi dan blockchain telah dimanfaatkan untuk memperkuat perlindungan serta keterlacakan catatan audit [6], [7]. Dalam konteks ini, hash berfungsi sebagai sidik digital yang merepresentasikan data; perubahan pada masukan akan menghasilkan nilai hash yang berbeda. Akan tetapi, penyimpanan hash hanya pada basis data yang sama dengan data operasional belum menyediakan pembanding yang independen apabila keduanya dapat diubah oleh pihak yang sama.

Pencatatan hash pada blockchain menawarkan lapisan pembanding yang terpisah dari basis data operasional. Sejumlah penelitian telah menerapkan blockchain dan smart contract untuk autentikasi dokumen perkantoran, penanda waktu, serta pelacakan riwayat perubahan [8]. Pendekatan pertukaran dokumen PDF berbasis blockchain juga menunjukkan penggunaan hash dokumen dan metadata sebagai bukti untuk memeriksa perubahan setelah dokumen dicatat [5]. Penerapan serupa telah dikembangkan untuk menghasilkan dan memverifikasi ijazah akademik digital [9], [10], mengelola sertifikat digital berbasis NFT [11], menjaga orisinalitas data pada proses manufaktur farmasi [12], serta memantau proses bisnis [13]. Studi mengenai sistem basis data hibrida turut membahas pemisahan penyimpanan operasional dan pencatatan berbasis blockchain beserta konsekuensi kinerjanya [14], [15].

Meskipun berbagai penelitian tersebut menunjukkan kegunaan blockchain untuk pencatatan bukti integritas, objek dan kebutuhan verifikasinya belum sepenuhnya sama dengan kuitansi sewa kos. Penelitian sistem pengelolaan kos lebih menekankan digitalisasi administrasi dan integrasi pembayaran [1], [2], [3], sedangkan penelitian verifikasi dokumen berbasis blockchain umumnya berfokus pada pencocokan hash dokumen atau metadata yang telah direkam [5], [8], [9], [10]. Pencocokan hash dapat menunjukkan bahwa data yang diperiksa berbeda dari data yang sebelumnya dicatat, tetapi hasil ketidakcocokan itu sendiri belum menjelaskan atribut mana yang berubah. Selain itu, blockchain membuktikan konsistensi terhadap data yang dicatat pada saat awal, bukan menjamin bahwa seluruh informasi yang dimasukkan sejak awal telah benar. Dengan demikian, masih diperlukan mekanisme yang menghubungkan status pembayaran terverifikasi, pencatatan hash kuitansi secara independen, dan pemeriksaan relasi data untuk membantu mendiagnosis atribut yang tidak konsisten.

Berdasarkan kebutuhan tersebut, penelitian ini mengembangkan mekanisme verifikasi integritas kuitansi digital sewa kos dengan arsitektur hibrida. Data operasional dan informasi pribadi tetap disimpan pada basis data MySQL di luar blockchain, sedangkan hash dari 17 atribut kuitansi dicatat pada smart contract di jaringan uji Ethereum Sepolia setelah notifikasi pembayaran Midtrans diverifikasi. Pemisahan ini menempatkan blockchain sebagai lapisan bukti integritas tambahan, bukan sebagai pengganti basis data, pengendalian akses, maupun audit internal [14], [15]. Sistem juga dilengkapi Diagnostic Tamper Engine yang memanfaatkan nilai pembanding dari relasi antartabel dan melakukan substitusi kandidat nilai untuk membantu mengidentifikasi atribut yang diduga berubah. Berbeda dari pemeriksaan hash yang hanya menghasilkan status cocok atau tidak cocok, mekanisme diagnosis ini dirancang untuk memberikan petunjuk atribut pada pola perubahan yang memiliki nilai pembanding relasional yang tersedia. Diagnosis spesifik tetap bergantung pada ketersediaan dan independensi nilai pembanding; perubahan beberapa atribut sekaligus atau perubahan pada nilai pembanding dapat menghasilkan diagnosis umum.

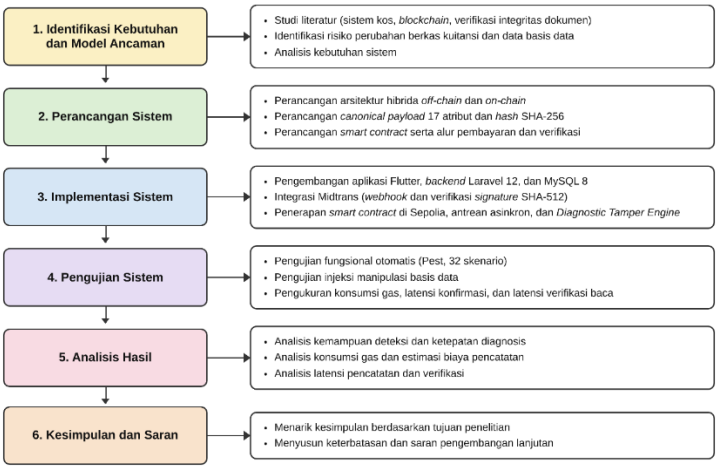
Penelitian ini bertujuan mengevaluasi mekanisme verifikasi integritas kuitansi digital sewa kos yang mengintegrasikan pemrosesan pembayaran Midtrans, hash anchoring pada Ethereum Sepolia, dan diagnosis ketidakcocokan berbasis relasi data. Evaluasi mencakup keberhasilan pencatatan hash, konsumsi gas, waktu konfirmasi pencatatan, latensi verifikasi baca, serta respons sistem terhadap skenario injeksi manipulasi basis data yang dirancang. Kontribusi penelitian terletak pada penggabungan bukti integritas kuitansi dengan

mekanisme diagnosis atribut berbasis relasi data pada konteks transaksi sewa kos. Lingkup klaim dibatasi pada pengujian di jaringan uji Sepolia dan skenario manipulasi terkontrol yang dilaporkan; penelitian ini tidak dimaksudkan untuk membuktikan kebenaran data sejak awal, mendeteksi seluruh kemungkinan manipulasi, atau menyimpulkan kinerja sistem pada jaringan produksi.

Penelitian ini bertujuan mengevaluasi mekanisme verifikasi integritas kuitansi digital sewa kos yang menggabungkan pemrosesan pembayaran Midtrans dan *hash anchoring* pada Ethereum. Evaluasi difokuskan pada kemampuan sistem mendeteksi perubahan atribut transaksi, keberhasilan pencatatan *hash* pada *blockchain*, konsumsi gas, waktu konfirmasi pencatatan, serta latensi verifikasi baca. Kontribusi penelitian berupa penggabungan bukti integritas kuitansi dengan mekanisme diagnosis berbasis relasi data, sehingga pemeriksaan tidak hanya menunjukkan ketidakcocokan *hash*, tetapi juga membantu mengenali jenis atribut yang berpotensi mengalami perubahan.

2. METODE PENELITIAN

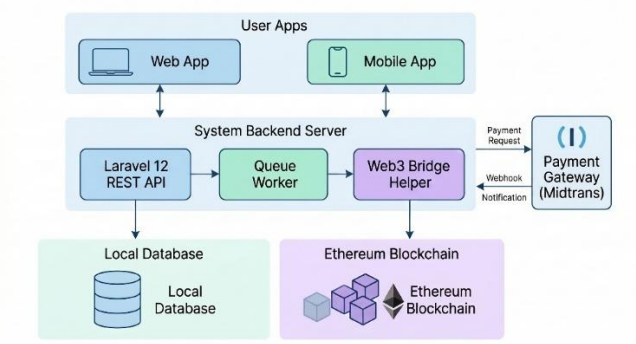
Penelitian ini menggunakan pendekatan rekayasa perangkat lunak untuk mengembangkan dan mengevaluasi mekanisme verifikasi integritas kuitansi digital sewa kos. Tahapan penelitian terdiri atas enam tahap, yaitu (1) identifikasi kebutuhan dan model ancaman, (2) perancangan sistem, (3) implementasi sistem, (4) pengujian sistem, (5) analisis hasil, serta (6) penarikan kesimpulan dan saran. Urutan tahapan tersebut ditunjukkan pada Gambar 1.



Gambar 1. Alur Penelitian

Berdasarkan Gambar 1, penelitian diawali dengan studi literatur, observasi pengelolaan data sewa Kos Pandu, dan identifikasi risiko perubahan data. Model ancaman dalam penelitian ini dibatasi pada perubahan atribut transaksi setelah pembayaran tercatat, terutama nominal pembayaran, tarif kamar, dan periode sewa. Penyuntingan berkas kuitansi digital menjadi konteks permasalahan, sedangkan eksperimen manipulasi dilakukan secara langsung terhadap data pada basis data operasional. Tahap perancangan menghasilkan arsitektur hibrida, struktur canonical JSON payload, rancangan smart contract, serta alur pembayaran dan verifikasi. Sistem kemudian diimplementasikan dan dievaluasi melalui pengujian fungsional otomatis, injeksi manipulasi basis data, dan pengukuran kinerja transaksi pada jaringan uji. Hak untuk mencatat data ke smart contract berada pada pengembang sistem, bukan administrator kos.

Data penelitian dikelompokkan menjadi tiga bagian. Pertama, data operasional sewa Kos Pandu yang meliputi data penghuni, kamar, tarif, tagihan, pembayaran, dan masa sewa. Data tersebut diperoleh melalui observasi terhadap sistem pengelolaan kos. Kedua, data transaksi pembayaran dan kinerja on-chain yang diperoleh melalui simulasi transaksi menggunakan payment gateway Midtrans serta pencatatan transaksi pada jaringan Ethereum Sepolia. Data on-chain mencakup identitas transaksi, nomor blok, status konfirmasi, konsumsi gas, dan biaya transaksi yang dapat ditelusuri melalui Etherscan. Ketiga, data pengujian integritas yang diperoleh melalui injeksi manipulasi atribut basis data SQL secara terkontrol dan pengujian fungsional otomatis menggunakan Pest PHP. Ketiga kelompok data tersebut digunakan untuk menelusuri alur pembayaran, memeriksa pencatatan bukti integritas, dan mengevaluasi respons sistem terhadap perubahan data.



Gambar 2. Arsitektur Hibrida Sistem

Gambar 2 memperlihatkan arsitektur hibrida yang memisahkan penyimpanan data operasional (*off-chain*) dari bukti integritas pada blockchain (*on-chain*). Pengguna mengakses sistem melalui aplikasi web atau mobile yang terhubung ke backend Laravel. Data penghuni, kamar, tagihan, pembayaran, masa sewa, dan informasi pribadi disimpan pada basis data MySQL. Setelah transaksi pembayaran terverifikasi dan perubahan data operasional berhasil disimpan, queue worker menjalankan proses pencatatan secara terpisah melalui komponen Web3 Bridge Helper. Komponen tersebut mengirimkan hash kuitansi ke smart contract pada jaringan uji Ethereum Sepolia. Smart contract menyimpan hash kuitansi, versi data, stempel waktu blok, dan alamat pencatat; data pribadi penghuni tidak disimpan pada blockchain. Pemisahan ini sejalan dengan pendekatan basis data hibrida yang memanfaatkan ledger sebagai bukti yang dapat diaudit dan diverifikasi [14], serta pendekatan penyimpanan data off-chain dengan pencatatan hash pada blockchain [15].

Bukti integritas kuitansi dibentuk menggunakan canonical JSON payload yang terdiri atas 17 atribut transaksi sebagaimana dirangkum pada Tabel 1. Sebelum proses hashing, nilai atribut dinormalisasi agar representasi data yang sama menghasilkan payload yang konsisten. Berdasarkan implementasi yang dijelaskan dalam penelitian ini, nilai nominal direpresentasikan dengan dua angka desimal, sedangkan tanggal dan waktu menggunakan format baku. Selanjutnya, pasangan kunci dan nilai diurutkan secara alfabetis berdasarkan nama kunci, kemudian diserialisasi menjadi string JSON secara deterministik sesuai Persamaan (1). Hash payload dihitung menggunakan SHA-256 [16] sesuai Persamaan (2). Hasil SHA-256 berukuran 32 byte atau 256 bit dan menjadi nilai yang dicatat pada smart contract.

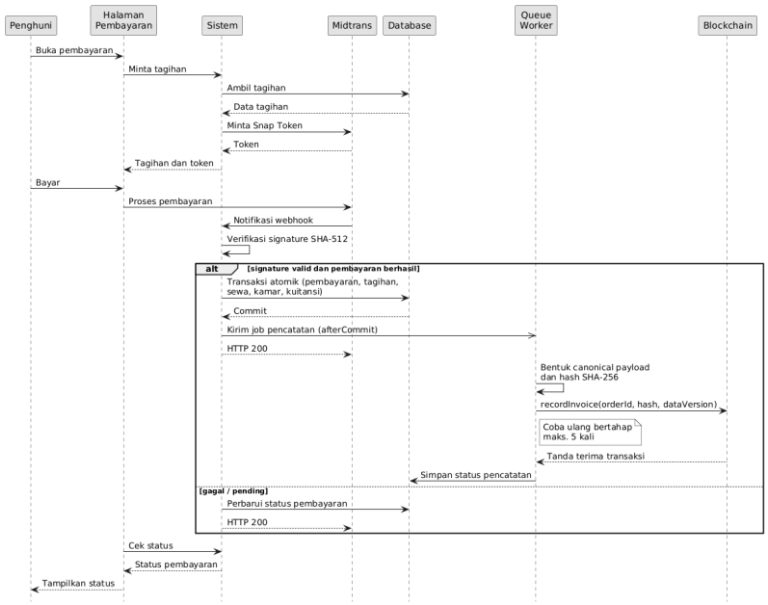
$$P_{\text{canonical}} = \text{Serialize}\big(\text{SortAlphabetical}(K, V)\big) \quad (1)$$

$$H = \text{"0x"} \parallel \text{SHA} - 256(P_{\text{canonical}}) \quad (2)$$

Pada Persamaan (1) dan (2), K menyatakan himpunan nama atribut, sedangkan V menyatakan himpunan nilai atribut setelah normalisasi. Fungsi SortAlphabetical mengurutkan pasangan kunci dan nilai berdasarkan nama kunci, sementara Serialize membentuk representasi JSON yang digunakan sebagai masukan algoritma SHA-256. Simbol ‖ menyatakan operasi penggabungan string. Dengan urutan atribut dan aturan representasi yang konsisten, payload yang sama dapat dihitung ulang untuk proses verifikasi.

Tabel 1. Atribut *canonical payload*

Kelompok	Atribut
Identitas transaksi	id_sewa, id_kamar, id_penghuni, nomor_kuitansi, order_id, payment_reference
Nilai transaksi	harga_per_bulan, nominal_bayar, total_tagihan, jumlah_bulan, jenis_tagihan
Periode dan waktu	tanggal_mulai, tanggal_selesai, tanggal_jatuh_tempo, tanggal_bayar
Metadata versi	schema_version, data_version



Gambar 3. Alur pembayaran dan pencatatan hash kuitansi

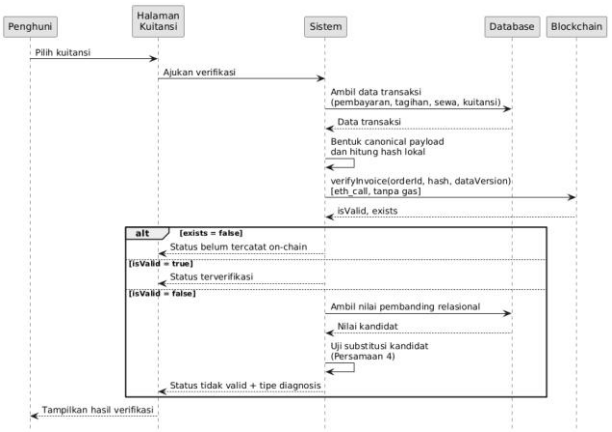
Gambar 3 menunjukkan alur pembayaran dan pencatatan hash kuitansi. Proses dimulai ketika penghuni memilih tagihan yang akan dibayar. Sistem mengambil informasi tagihan dari basis data dan meminta Snap Token kepada Midtrans untuk memulai proses pembayaran. Setelah penghuni menyelesaikan pembayaran, Midtrans mengirimkan notifikasi webhook ke server. Server memeriksa keabsahan notifikasi dengan menghitung signature menggunakan Persamaan (3), kemudian membandingkan hasil perhitungan tersebut dengan signature yang diterima. Notifikasi hanya diteruskan ke proses pembaruan transaksi apabila kedua signature sesuai.

$$S_{\text{calc}} = \text{SHA} - 512(\text{OrderID} \parallel \text{StatusCode} \parallel \text{GrossAmount} \parallel \text{ServerKey}) \qquad (3)$$

Pada Persamaan (3), OrderID, StatusCode, dan GrossAmount merupakan nilai yang diterima dari notifikasi Midtrans, sedangkan ServerKey merupakan kunci rahasia merchant yang digunakan oleh server untuk menghitung signature. ServerKey disimpan sebagai informasi rahasia dan tidak ditampilkan kepada pengguna.

Pemrosesan notifikasi dirancang secara idempoten untuk menangani kemungkinan notifikasi terkirim berulang, terlambat, atau tidak berurutan. Dengan demikian, satu pembayaran tidak menghasilkan kuitansi atau perpanjangan sewa ganda. Setelah signature dinyatakan sesuai, pembaruan status pembayaran, tagihan, sewa, dan kamar serta penerbitan kuitansi dilakukan dalam satu transaksi basis data atomik. Pencatatan hash ke blockchain dijalankan secara asinkron melalui antrean setelah transaksi basis data berhasil disimpan. Pemisahan proses ini memungkinkan status pembayaran tetap tersimpan ketika koneksi ke jaringan Ethereum mengalami gangguan. Jika pencatatan blockchain gagal, sistem melakukan percobaan ulang secara bertahap hingga lima kali dan memperbarui status pencatatan setelah transaksi on-chain terkonfirmasi.

Smart contract menyediakan dua fungsi utama, yaitu pencatatan dan verifikasi. Fungsi pencatatan menyimpan hash kuitansi beserta versi datanya dan hanya dapat dipanggil oleh pemilik kontrak. Fungsi verifikasi membaca hash yang telah tersimpan dan membandingkannya dengan hash yang dikirim oleh aplikasi, sehingga pemeriksaan baca tidak memerlukan transaksi penulisan baru. Setiap versi data dicatat secara terpisah bersama stempel waktu blok agar riwayat pencatatan dapat ditelusuri. Pemanfaatan blockchain sebagai jejak audit juga telah dibahas sebagai pendekatan untuk memperkuat keterlacakan catatan audit [7].



Gambar 4. Alur verifikasi dan diagnosis kuitansi

Berdasarkan Gambar 4, verifikasi dimulai dengan mengambil data pembayaran, tagihan, sewa, dan kuitansi dari basis data. Sistem membentuk kembali *canonical* JSON *payload* dari data tersebut dan menghitung hash lokal menggunakan prosedur yang sama dengan proses pencatatan. Hash lokal kemudian dibandingkan dengan hash yang tersimpan pada smart contract. Jika catatan belum tersedia di blockchain, sistem menampilkan status belum tercatat *on-chain*. Jika catatan tersedia dan kedua hash sama, kuitansi dinyatakan terverifikasi. Jika hash berbeda, kuitansi dinyatakan tidak valid dan sistem menjalankan *Diagnostic Tamper Engine*. Perbandingan hash data *off-chain* dengan catatan on-chain juga digunakan pada rancangan penyimpanan data multipenyewa dan verifikasi integritas ijazah digital pada jaringan Sepolia [15].

Diagnostic Tamper Engine menggunakan redundansi relasional antartabel untuk membantu menelusuri atribut yang diduga berubah, sebagaimana dirangkum pada Algoritma 1. Untuk setiap atribut yang diperiksa, sistem mengambil nilai kandidat dari entitas atau atribut lain yang memiliki hubungan dengan data tersebut. Sebagai contoh, tanggal selesai sewa dapat diperiksa melalui tanggal mulai dan jumlah bulan sewa atau dibandingkan dengan tanggal jatuh tempo tagihan perpanjangan. Tarif kamar diperiksa menggunakan nilai tarif pada data kamar atau tagihan, sedangkan nominal pembayaran dibandingkan dengan total tagihan. Setiap kandidat kemudian disubstitusikan ke atribut terkait pada payload lokal, dan hash dari payload hasil substitusi dihitung sesuai Persamaan (4).

$$H_{\text{uji}}(a, c) = \text{"0x"} \parallel \text{SHA} - 256(P_{\text{lokal}}[a \leftarrow c]) \qquad (4)$$

Pada Persamaan (4), P_lokal[a ← c] menyatakan payload lokal setelah nilai atribut a diganti dengan nilai kandidat c. Jika hash hasil substitusi sama dengan hash on-chain, sistem menandai atribut tersebut sebagai kandidat perubahan dan mengembalikan tipe diagnosis yang sesuai. Jika tidak ada substitusi tunggal yang menghasilkan hash yang sama, sistem mengembalikan diagnosis umum lease_data_mismatch. Kondisi ini dapat terjadi, antara lain, ketika beberapa atribut berubah secara bersamaan atau nilai pembanding ikut berubah. Oleh sebab itu, diagnosis spesifik bergantung pada ketersediaan, ketepatan, dan independensi nilai pembanding yang digunakan.

Selain mencocokkan hash kuitansi, sistem memeriksa konsistensi tanggal jatuh tempo pada tagihan perpanjangan yang belum dibayar. Atribut tanggal jatuh tempo tagihan perpanjangan tidak termasuk dalam payload kuitansi yang telah dicatat sehingga perubahan pada atribut tersebut tidak mengubah hash kuitansi. Untuk pemeriksaan ini, setelah hash kuitansi terakhir dinyatakan cocok dengan catatan on-chain, tanggal selesai sewa pada kuitansi digunakan sebagai acuan untuk memeriksa tanggal jatuh tempo tagihan perpanjangan berikutnya. Jika kedua tanggal tidak sesuai, sistem memberikan diagnosis due_date_mismatch. Pemeriksaan tambahan ini dimaksudkan untuk menemukan ketidaksesuaian data terkait kuitansi yang berada di luar atribut payload yang di-hash.

Algoritma 1. Diagnosis ketidakcocokan kuitansi

Masukan: data transaksi lokal T, status keberadaan catatan E, hash on-chain H_on
Keluaran: status verifikasi dan tipe diagnosis
1. P_lokal ← BentukPayload(T)
2. Jika E = false, kembalikan (belum tercatat on-chain, -)
3. Jika Hash(P_lokal) = H_on, kembalikan (terverifikasi, -)
4. D ← daftar atribut dan tipe diagnosis yang didukung
5. Untuk setiap atribut a dan tipe diagnosis dalam D
6. C ← AmbilKandidat(a, T)
7. Untuk setiap kandidat c dalam C

8. Jika Hash(P_lokal[a ← c]) = H_on, kembalikan (tidak valid, tipe diagnosis)
9. Kembalikan (tidak valid, lease_data_mismatch)

Pengujian dilakukan dalam tiga kelompok. Kelompok pertama mengukur kinerja pencatatan dan verifikasi pada jaringan Ethereum Sepolia. Pengukuran pencatatan mencakup enam transaksi hash, konsumsi gas, biaya transaksi, dan latensi konfirmasi. Data identitas transaksi, nomor blok, status, serta gas yang digunakan diperoleh dari catatan transaksi pada Etherscan. Konsumsi gas digunakan sebagai ukuran sumber daya eksekusi smart contract, sebagaimana diterapkan dalam kajian biaya gas Ethereum [17]. Latensi konfirmasi dihitung sebagai selisih waktu antara dimulainya proses pencatatan dan diterimanya konfirmasi transaksi oleh sistem. Latensi verifikasi diukur dari waktu respons pemanggilan fungsi baca pada smart contract sebanyak sepuluh kali berturut-turut. Biaya transaksi dihitung dari hasil kali jumlah gas yang digunakan dan harga gas per unit [18], sebagaimana dirumuskan pada Persamaan (5). Karena harga gas pada jaringan uji tidak merepresentasikan biaya pada jaringan utama Ethereum, biaya jaringan produksi diperlakukan sebagai estimasi terpisah.

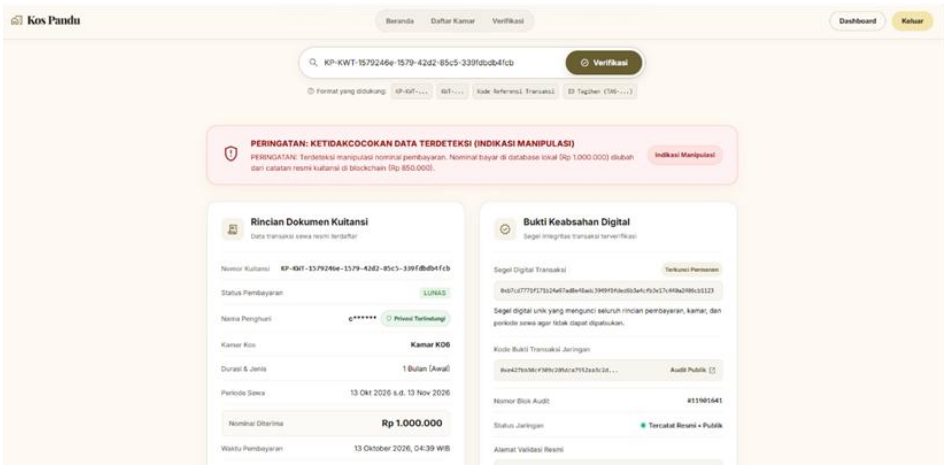
$$Fee = GasUsed \times GasPrice \quad (5)$$

Pada Persamaan (5), GasUsed menyatakan jumlah gas yang digunakan oleh transaksi pencatatan, sedangkan GasPrice menyatakan harga gas per unit dalam satuan ETH. Apabila biaya dikonversi ke rupiah, nilai tersebut merupakan estimasi yang bergantung pada asumsi harga gas dan kurs ETH terhadap rupiah pada waktu perhitungan.

Kelompok kedua merupakan pengujian injeksi manipulasi basis data secara terkontrol. Pengujian mencakup satu kasus kontrol tanpa perubahan dan empat kasus manipulasi, dengan mengubah satu atribut pada setiap skenario, yaitu nominal pembayaran, tanggal selesai sewa, tarif per bulan, atau tanggal jatuh tempo tagihan perpanjangan. Setelah setiap skenario diuji, nilai atribut dikembalikan ke kondisi semula sebelum pengujian berikutnya dilakukan. Sistem dievaluasi berdasarkan status verifikasi dan tipe diagnosis yang ditampilkan. Kelompok ketiga merupakan pengujian fungsional otomatis menggunakan Pest pada framework Laravel. Sebanyak 32 skenario uji dibagi ke dalam empat modul, yaitu determinisme hash, idempotensi notifikasi Midtrans, pencatatan dan verifikasi on-chain, serta Diagnostic Tamper Engine. Setiap skenario dinilai berdasarkan kesesuaian keluaran sistem terhadap kondisi dan hasil yang diharapkan pada kasus uji.

3. HASIL DAN ANALISIS

Sistem verifikasi kuitansi digital diimplementasikan pada aplikasi web dan mobile serta dihubungkan dengan smart contract KosPanduIntegrity pada jaringan uji Ethereum Sepolia. Hasil pengujian difokuskan pada keberhasilan pencatatan dan verifikasi hash, kemampuan mendeteksi perubahan data, serta keberhasilan fungsi utama sistem. Dua tampilan berikut disajikan sebagai prototipe antarmuka sistem, bukan sebagai bukti kuantitatif pengujian.



Gambar 5. Prototipe halaman verifikasi kuitansi pada aplikasi web



Gambar 6. Prototipe tampilan verifikasi integritas kuitansi pada aplikasi mobile

Gambar 5 dan Gambar 6 memperlihatkan dua antarmuka untuk memeriksa kuitansi yang sama. Pada halaman web, pengguna dapat memasukkan nomor kuitansi untuk melihat rincian transaksi dan status integritasnya. Aplikasi mobile menampilkan informasi verifikasi serta akses untuk memeriksa bukti transaksi pada jaringan Sepolia. Nama penghuni ditampilkan secara tersamar pada tampilan publik. Kedua gambar menunjukkan implementasi antarmuka; penilaian keberhasilan sistem tetap didasarkan pada hasil pengujian terukur yang dijelaskan berikut.

Pengujian pencatatan hash dilakukan terhadap enam transaksi pada jaringan *Ethereum Sepolia*. Seluruh transaksi berhasil dikonfirmasi (6 dari 6 transaksi atau 100%), dengan konsumsi gas yang sama, yaitu 576.856 gas per transaksi. Latensi konfirmasi pencatatan memiliki rata-rata 14,00 detik, dengan rentang 8–17 detik. Sementara itu, pengukuran sepuluh pemanggilan fungsi verifikasi baca menghasilkan latensi rata-rata 277,0 ms, dengan rentang 251,2–325,7 ms. Fungsi verifikasi hanya membaca data pada smart contract sehingga tidak memerlukan transaksi penulisan dan biaya gas.

Tabel 3 merangkum indikator kinerja utama. Nilai keberhasilan pencatatan 100% menunjukkan bahwa keenam transaksi yang diamati berhasil dikonfirmasi pada jaringan uji, bukan merupakan ukuran kinerja pada beban transaksi tinggi. Demikian pula, nilai latensi merupakan hasil pengukuran pada lingkungan pengujian yang digunakan dan dapat berbeda pada jaringan produksi.

Tabel 3. Ringkasan hasil pengujian kinerja dan fungsional

Indikator	Hasil pengujian	Jumlah/ulangan	Keterangan
Keberhasilan pencatatan on-chain	100%	6/6 transaksi	Seluruh transaksi berstatus Success
Konsumsi gas	576.856 gas	6 transaksi	Konstan pada seluruh transaksi
Latensi konfirmasi pencatatan	14,00 detik	6 transaksi	Rentang 8–17 detik
Latensi verifikasi baca	277,0 ms	10 pemanggilan	Rentang 251,2–325,7 ms
Pengujian fungsional	100%	32/32 skenario	Seluruh skenario dinyatakan lolos

Pengujian injeksi manipulasi dilakukan dengan mengubah satu atribut pada setiap skenario, kemudian mengembalikannya ke nilai semula sebelum skenario berikutnya dijalankan. Empat skenario manipulasi yang diuji meliputi perubahan nominal pembayaran, tanggal selesai sewa, tarif per bulan, dan tanggal jatuh tempo tagihan perpanjangan. Seluruh perubahan terdeteksi (4 dari 4 skenario atau 100% tingkat keberhasilan deteksi pada skenario uji). Satu skenario kontrol tanpa manipulasi juga menghasilkan status terverifikasi. Hasil tersebut dirangkum pada Tabel 4.

Tabel 4. Ringkasan hasil pengujian injeksi manipulasi

Skenario	Atribut yang diubah	Status sistem	Diagnosis
A	Nominal pembayaran	Tidak valid	payment_amount_mismatch
B	Tanggal selesai sewa	Tidak valid	lease_date_mismatch
C	Tarif per bulan	Tidak valid	lease_rate_mismatch
D	Tanggal jatuh tempo tagihan perpanjangan	Tidak valid	due_date_mismatch
Kontrol	Tidak ada perubahan	Terverifikasi	–

Pada tiga skenario pertama, perubahan atribut menyebabkan hash lokal berbeda dari hash yang tersimpan pada blockchain. Diagnostic Tamper Engine kemudian menggunakan nilai pembanding dari relasi antartabel untuk menunjukkan atribut yang diduga berubah. Pada skenario perubahan tanggal jatuh tempo tagihan

perpanjangan, hash kuitansi tidak berubah karena atribut tersebut berada di luar payload kuitansi. Ketidaksesuaian tetap ditemukan melalui pemeriksaan relasional dengan menggunakan tanggal selesai sewa pada kuitansi terverifikasi sebagai acuan. Dengan demikian, mekanisme pemeriksaan mencakup pencocokan hash dan pemeriksaan konsistensi atribut terkait yang tidak termasuk dalam payload.

Secara keseluruhan, 32 dari 32 skenario pengujian fungsional berhasil dilalui (100%). Pengujian tersebut mencakup determinisme hash, idempotensi notifikasi Midtrans, pencatatan dan verifikasi on-chain, serta fungsi Diagnostic Tamper Engine. Jika istilah akurasi digunakan untuk menggambarkan hasil deteksi, angka 100% pada penelitian ini hanya merujuk pada keberhasilan mendeteksi empat skenario manipulasi yang telah dirancang. Nilai tersebut bukan akurasi klasifikasi yang diperoleh dari dataset uji berskala besar dan tidak dapat digeneralisasi untuk seluruh jenis manipulasi.

Hasil ini menunjukkan bahwa prototipe dapat menjalankan fungsi pencatatan dan pemeriksaan integritas kuitansi pada jaringan uji serta mendeteksi pola perubahan data yang dicakup dalam skenario pengujian. Namun, jumlah transaksi on-chain dan skenario manipulasi masih terbatas. Pengujian belum mencakup perubahan beberapa atribut secara bersamaan, manipulasi terhadap nilai pembanding, maupun evaluasi beban tinggi pada jaringan produksi. Oleh karena itu, hasil yang diperoleh lebih tepat dipahami sebagai bukti keberhasilan fungsi pada skenario terkontrol, bukan sebagai jaminan keamanan menyeluruh.

4. KESIMPULAN

Penelitian ini menghasilkan mekanisme verifikasi integritas kuitansi digital sewa kos yang mengintegrasikan *Midtrans*, hash SHA-256, dan smart *contract Ethereum Sepolia*, serta dilengkapi *Diagnostic Tamper Engine* berbasis relasi antartabel. Berdasarkan pengujian terkontrol, sistem berhasil mendeteksi dan mengidentifikasi perubahan pada seluruh empat skenario manipulasi (4/4), sehingga tingkat keberhasilan deteksi pada skenario yang diuji mencapai 100%; sementara skenario kontrol tanpa perubahan tetap dinyatakan terverifikasi. Seluruh 32 skenario pengujian fungsional juga berhasil (32/32 atau 100%), dan enam transaksi pencatatan hash berhasil dikonfirmasi (6/6 atau 100%). Dari sisi kinerja, pencatatan menggunakan 576.856 gas per kuitansi dengan latensi konfirmasi rata-rata 14,00 detik, sedangkan verifikasi baca memiliki latensi rata-rata 277,0 ms. Hasil tersebut menunjukkan bahwa mekanisme dapat menjalankan verifikasi integritas serta diagnosis perubahan data pada pola manipulasi yang diuji. Namun, angka 100% merupakan tingkat keberhasilan pada empat skenario terkontrol, bukan akurasi universal terhadap seluruh bentuk manipulasi. Pengujian masih terbatas pada jaringan Sepolia, data simulasi, dan perubahan satu atribut pada setiap skenario; karena itu, pengujian lanjutan pada data operasional, perubahan beberapa atribut sekaligus, serta kondisi jaringan dan beban transaksi yang lebih beragam masih diperlukan.

UCAPAN TERIMA KASIH

Penulis mengucapkan terima kasih kepada Tuhan Yang Maha Esa, orang tua atas doa dan dukungannya, serta dosen pembimbing atas bimbingan dan arahan selama penelitian dan penyusunan artikel ini. Terima kasih juga disampaikan kepada institusi tempat penulis menempuh pendidikan, pengelola Kos Pandu sebagai tempat observasi, teman dan sahabat yang senantiasa memberikan semangat, serta semua pihak yang telah membantu hingga penelitian ini dapat diselesaikan.

REFERENSI

[1] F. F. Azzahra, B. R. Sekamayang, and R. Habibi, “Integrasi Google OAuth dan Payment Gateway Midtrans dalam KosConnect,” *J. Process.*, vol. 20, no. 1, 2025, doi: 10.33998/processor.2025.20.1.2185.

[2] S. Alfarezy, M. F. Ridho, and J. Prayoga, “Sistem Pembayaran Kos Berbasis Web Menggunakan Midtrans Payment Gateway,” *Syntax J. Softw. Eng. Comput. Sci. Inf. Technol.*, vol. 5, no. 2, pp. 580–585, 2024, doi: 10.46576/syntax.v5i2.5558.

[3] A. Bintang, I. Purnamasari, and Carudin, “Transformasi Digital Pengelolaan Kos: Sistem Berbasis Laravel dengan Integrasi Pembayaran Nontunai,” *J. Inform. dan Tek. Elektro Terap.*, vol. 14, no. 1, 2026, doi: 10.23960/jitet.v14i1.8261.

[4] S. Boonkrong, “Design of an academic document forgery detection system,” *Int. J. Inf. Technol.*, vol. 17, pp. 5175–5187, 2025, doi: 10.1007/s41870-024-02006-6.

[5] P. Mishra and R. Ganesan, “Securing E-governance: A blockchain-based framework for tamper-proof PDF document exchange,” *Front. Blockchain*, vol. 8, 2025, doi: 10.3389/fbloc.2025.1699773.

[6] L. Gao, “Enterprise internal audit data encryption based on blockchain technology,” *PLoS One*, vol.

20, no. 1, 2025, doi: 10.1371/journal.pone.0315759.

[7] C. Regueiro, I. Seco, I. Gutiérrez-Agüero, B. Urquizu, and J. Mansell, "A blockchain-based audit trail mechanism: Design and implementation," *Algorithms*, vol. 14, no. 12, 2021, doi: 10.3390/a14120341.

[8] X. Zhai, S. Pang, M. Wang, S. Qiao, and Z. Lv, "TVS: A trusted verification scheme for office documents based on blockchain," *Complex Intell. Syst.*, vol. 9, no. 3, pp. 2865–2877, 2023, doi: 10.1007/s40747-021-00617-1.

[9] A. Rustemi, F. Dalipi, V. Atanasovski, and A. Risteski, "DIAR: A blockchain-based system for generation and verification of academic diplomas," *Discov. Appl. Sci.*, vol. 6, 2024, doi: 10.1007/s42452-024-05984-1.

[10] R. B. Bimantoro, H. E. Wahanani, and M. M. Al Haromainy, "Perancangan dan implementasi smart contract untuk verifikasi integritas ijazah digital pada jaringan blockchain," *Jutisi J. Ilm. Tek. Inform. dan Sist. Inf.*, vol. 15, no. 3, pp. 1309–1322, 2026, doi: 10.35889/jutisi.v15i3.3741.

[11] M. I. Sumita and R. I. Akbar, "Design and build of an NFT-based digital certificate management application on the Solana network," *JSAI J. Sci. Appl. Informatics*, vol. 8, no. 2, 2025, doi: 10.36085/jsai.v8i2.8321.

[12] M. Durá and et al., "Blockchain for data originality in pharma manufacturing," *J. Pharm. Innov.*, vol. 18, no. 4, pp. 1745–1763, 2023, doi: 10.1007/s12247-023-09748-z.

[13] C. Di Ciccio, G. Meroni, and P. Plebani, "On the adoption of blockchain for business process monitoring," *Softw. Syst. Model.*, vol. 21, pp. 915–937, 2022, doi: 10.1007/s10270-021-00959-x.

[14] Z. Ge, D. Loghin, B. C. Ooi, P. Ruan, and T. Wang, "Hybrid blockchain database systems: Design and performance," *Proc. VLDB Endow.*, vol. 15, no. 5, pp. 1092–1104, 2022, doi: 10.14778/3510397.3510406.

[15] A. Sharma and P. Kaur, "Tamper-proof multitenant data storage using blockchain," *Peer-to-Peer Netw. Appl.*, vol. 16, no. 1, pp. 431–449, 2023, doi: 10.1007/s12083-022-01410-8.

[16] M. Ridwan, R. Umar, M. K. Biddinika, and P. W. Prasetyo, "Experimental evaluation of avalanche effect and hash consistency in SHA-256, SHA3-256, and BLAKE2b-256 for smart grid data integrity verification," *J. Inf. Syst. Informatics*, vol. 8, no. 4, pp. 4184–4210, 2026, doi: 10.63158/journalisi.v8i4.1685.

[17] M. T. Ta and T. Q. Do, "A study on gas cost of Ethereum smart contracts and performance of blockchain on simulation tool," *Peer-to-Peer Netw. Appl.*, vol. 17, no. 1, pp. 200–212, 2024, doi: 10.1007/s12083-023-01598-3.

[18] B. K. Meister and H. C. W. Price, "Gas fees on the Ethereum blockchain: From foundations to derivative valuations," *Front. Blockchain*, vol. 7, 2024, doi: 10.3389/fbloc.2024.1462666.