IMPLEMENTASI JSON WEB TOKEN UNTUK AUTENTIKASI DAN OTORISASI PADA SISTEM PEMBAYARAN DIGITAL MENGGUNAKAN FRAMEWORK FLASK

Raffles Agustinus Supit, Yeremia Alfa Susetyo
Teknik Informatika, Universitas Kristen Satya Wacana
Jl. Dr. O. Notohamidjojo No 1-10, Blotongan, Kec. Sidorejo, Kota Salatiga 50715
672022130@student.uksw.edu

ABSTRAK

Penelitian berfokus pada upaya peningkatan keamanan pada sistem pembayaran digital, khususnya dalam hal autentikasi dan otorisasi *Application Programming Interface* (API) yang kerap menjadi sasaran serangan siber. Metode yang digunakan adalah pengembangan sistem dengan mengintegrasikan *JSON Web Token* (JWT) ke dalam *framework* Flask untuk memperkuat mekanisme autentikasi dan otorisasi API. JWT dipilih karena bersifat stateless, ringan, dan mendukung klaim terstruktur, sehingga cocok untuk sistem API yang membutuhkan skalabilitas tinggi. Flask dipilih karena arsitekturnya yang minimalis dan modular memudahkan integrasi komponen keamanan secara terpisah tanpa overhead yang tidak diperlukan. Berdasarkan pengujian terhadap empat aspek, yaitu keamanan JWT, keamanan API, fungsionalitas transaksi, dan *concurrency*, sistem menunjukkan tingkat keberhasilan 100% pada seluruh 4 kasus uji validasi token JWT dan 8 skenario fungsionalitas transaksi. Pengujian keamanan API berdasarkan OWASP mencakup 6 kategori kerentanan dan seluruhnya lulus. Pengujian concurrency dengan 10 permintaan *top-up* bersamaan menghasilkan saldo akhir yang konsisten sebesar Rp500.000 tanpa duplikasi transaksi. Hasil pengujian menunjukkan bahwa penggunaan JWT dapat meningkatkan keamanan dan efisiensi manajemen akses, serta mengurangi risiko penyalahgunaan API. Kesimpulan penelitian menegaskan bahwa integrasi JWT dengan strategi yang tepat dapat menjadi fondasi utama dalam membangun sistem pembayaran digital yang aman dan terstandarisasi.

Kata kunci : *API, JWT, Flask, Sistem Pembayaran Digital*

1. PENDAHULUAN

Perkembangan teknologi informasi di Indonesia telah mendorong transformasi yang signifikan di berbagai sektor, terutama sektor keuangan. Salah satu dampak yang paling terlihat adalah pergeseran metode pembayaran dari sistem konvensional berbasis uang tunai menjadi sistem pembayaran digital. Penggunaan dompet digital (*e-wallet*), *mobile banking*, dan platform *fintech* lainnya semakin diminati oleh masyarakat karena menawarkan kemudahan, kecepatan, dan efisiensi dalam bertransaksi [1].

Seiring dengan pesatnya perkembangan teknologi informasi di kalangan masyarakat, terjadi perubahan besar dalam cara bertransaksi, di mana saat ini banyak orang lebih memilih metode pembayaran digital karena dapat menghemat waktu dan menawarkan sistem penyimpanan informasi yang aman [2].

Hal ini menunjukkan bahwa sistem pembayaran di Indonesia secara bertahap mulai bergeser dari metode konvensional menjadi digital [3].

Seiring dengan meningkatnya minat masyarakat untuk menggunakan sistem pembayaran digital, terdapat berbagai masalah krusial yang mengancam keamanan dan integritas infrastruktur pembayaran. Salah satu masalah utamanya adalah kurangnya autentikasi *Application Programming Interface* (API), yang merupakan sarana komunikasi antar sistem pembayaran. Pada praktiknya, banyak sistem masih menggunakan autentikasi dasar atau kunci API tanpa enkripsi yang memadai, sehingga rentan terhadap pencurian identitas dan akses yang tidak sah [4].

Kondisi ini secara langsung membuka celah bagi pihak tidak berwenang untuk mengeksploitasi sistem, yang pada akhirnya dapat mengakibatkan kebocoran data transaksi dan kerugian finansial bagi pengguna. Selain itu, kurangnya ketegasan dalam otorisasi akses API membuat sistem rentan diretas oleh pihak yang tidak berkepentingan, sehingga membuka peluang penyalahgunaan data transaksi pengguna. Hal tersebut diperparah dengan rendahnya kesadaran pengguna dalam menjaga kerahasiaan data pribadi yang seringkali menjadi sasaran empuk serangan siber, seperti phising [5].

Di sisi lain, adanya keragaman sistem dan belum adanya standarisasi integrasi dalam layanan pembayaran digital dapat menyebabkan integrasi yang tidak konsisten serta meningkatkan risiko inefisiensi operasional dan kebocoran data [6]].

Kurangnya integrasi dalam manajemen API dan sistem otorisasi juga melemahkan pertahanan terhadap potensi serangan siber, sehingga sistem menjadi lebih mudah dieksploitasi secara berulang tanpa terdeteksi. Fenomena yang terjadi menunjukkan adanya kebutuhan mendesak untuk memperkuat sistem autentikasi dan otorisasi, seperti melalui penerapan autentikasi yang terbukti dapat mengurangi risiko akses ilegal dan meningkatkan kepercayaan pengguna [7].

Sebagai upaya dalam mengatasi permasalahan keamanan sistem pembayaran digital, diperlukan autentikasi dan otorisasi yang jelas. Salah satu teknologi yang dapat diimplementasikan untuk melakukan proses autentikasi dan otorisasi tersebut

adalah *JSON Web Token* (JWT). Dengan menggunakan JWT keamanan sistem dapat ditingkatkan karena token yang dihasilkan akan diverifikasi terlebih dahulu oleh server sebelum memberikan akses ke API, sehingga hanya pengguna dengan token yang valid yang dapat mengakses sumber daya sistem [8].

Namun, keamanan penggunaan JWT sangat bergantung pada manajemen token yang tepat. Pengelolaan token yang aman seperti mekanisme logout yang memblokir token yang lama dapat mencegah penyalahgunaan token yang telah kedaluwarsa atau dicuri [9].

Di sisi lain, diperlukan strategi pencabutan token JWT untuk menonaktifkan token yang sudah tidak berlaku [10]. Oleh karena itu, diperlukan pendekatan yang komprehensif dalam penerapan JWT agar sistem autentikasi dan otorisasi tetap aman dari ancaman keamanan.

Selain menggunakan teknologi JWT sebagai pengelola autentikasi dan otorisasi, diperlukan sebuah layanan yang dapat digunakan untuk mengembangkan aplikasi web. Flask merupakan salah satu kerangka kerja pengembangan aplikasi web yang dapat digunakan. Penggunaan Flask memungkinkan untuk membuat layanan berbasis web yang dapat diakses melalui protokol HTTP, memiliki kemampuan untuk berbagi data dan berinteraksi dengan sistem, serta memungkinkan pemecahan sistem menjadi *service* yang modular sehingga cocok untuk digunakan dalam sistem pembayaran digital [11].

Dengan kerentanan sistem pembayaran digital terutama dalam keamanan data dan pembagian akses, maka mengkombinasikan JWT dengan Flask menjadi solusi dalam meningkatkan keamanan dalam layanan sistem pembayaran digital. Flask yang memungkinkan membuat layanan web ditambah dengan token yang dihasilkan dan divalidasi oleh JWT, kedua teknologi dapat meminimalkan penyalahgunaan akses terhadap API.

Penelitian yang dilakukan diharapkan dapat meningkatkan efisiensi kinerja sistem melalui penggunaan token yang mampu mempercepat proses autentikasi serta mendukung skalabilitas layanan pembayaran digital. Hal tersebut sangat mungkin karena JWT bersifat ringan, tidak memerlukan penyimpanan sesi di server, dan kompatibel dengan API, sehingga sistem dapat berkembang dengan lebih fleksibel.

2. TINJAUAN PUSTAKA

Penelitian terdahulu yang berjudul "Implementasi API Restful dengan *JSON Web Token* (JWT) Pada Aplikasi E-Commerce Thrifty Shop untuk Otentikasi dan Otorisasi Pengguna" menunjukkan bahwa penggunaan JWT pada sistem *e-commerce* mampu memberikan hasil yang efektif dalam menjaga keamanan proses autentikasi serta validitas data barang [8].

Penelitian tersebut mengimplementasikan *JSON Web Token* (JWT) dengan algoritma HMAC SHA-256 pada sistem web service guna mengamankan proses autentikasi dan otorisasi pengguna. JWT dimanfaatkan sebagai metode autentikasi berbasis token yang menghasilkan token terenkripsi untuk memvalidasi setiap permintaan ke API. Verifikasi token dilakukan secara otomatis melalui *middleware* yang aktif dalam setiap interaksi antara klien dan server. Sistem dikembangkan menggunakan arsitektur RESTful API dengan Express.js sebagai *framework backend*, di mana token JWT dihasilkan setelah proses *login* berhasil dan dikirimkan kembali melalui *header Authorization* dalam bentuk *Bearer Token* untuk setiap permintaan berikutnya. Hasil pengujian menunjukkan bahwa sistem mampu melindungi *endpoint* API dengan rata-rata waktu respons sebesar 151,75 ms, menolak permintaan tanpa token valid dengan memberikan respons "*Unauthorized Access API*", serta menjamin integritas data melalui proses *signing* dan verifikasi token menggunakan *secret key* yang tersimpan secara aman dalam *environment variables*. Dengan adanya penelitian terdahulu tersebut maka mengimplementasikan JWT dalam sistem pembayaran digital merupakan salah satu solusi untuk meningkatkan keamanan dalam proses autentikasi dan otorisasi.

Berdasarkan penelitian yang berjudul "Implementasi Python API dengan Flask Framework sebagai Cloud Run Service untuk Proses Update di PT.XYZ" menjelaskan bahwa penggunaan *framework* Flask pada sistemnya dapat mempercepat proses pembuatan API [12].

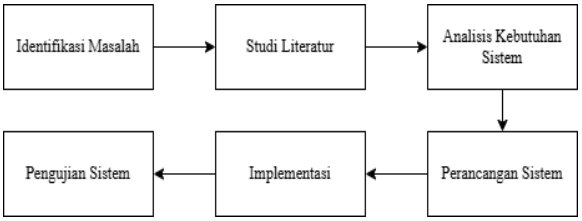
Penelitian tersebut menemukan bahwa Flask merupakan *framework* yang ringan dan fleksibel. Kemampuan Flask untuk beradaptasi dengan berbagai kebutuhan teknologi tercermin dari keberhasilannya diintegrasikan ke dalam Google Cloud Platform (GCP), dukungan terhadap kontainerisasi menggunakan Docker, serta kemampuannya membangun arsitektur modular yang memisahkan fungsi-fungsi seperti *hashing* dan validasi JSON. Sebagai *framework* yang sederhana namun fungsional, Flask memungkinkan pengembangan sistem keamanan dan validasi yang dapat disesuaikan dengan kebutuhan spesifik bisnis, sehingga memberikan kebebasan dalam pengelolaan alur autentikasi. Dari hasil studi tersebut yang menunjukkan kelebihan Flask dalam hal kesederhanaan, fleksibilitas, dan skalabilitas menjadikannya *framework* yang relevan untuk digunakan dalam pengembangan API pada sistem pembayaran digital.

Penelitian terdahulu terkait autentikasi berbasis JWT dan penggunaan framework Flask pada layanan *cloud* telah membuktikan efektivitas pendekatan keamanan dalam mengatasi tantangan integrasi dan validasi data. Namun, masih terdapat kekurangan dalam penerapan standar keamanan yang komprehensif, khususnya untuk sistem pembayaran digital yang memerlukan autentikasi dan otorisasi

yang dinamis dan berlapis. Oleh karena itu, penelitian ini mengimplementasikan JWT untuk autentikasi dan otorisasi pada sistem pembayaran digital menggunakan Flask.

3. METODE PENELITIAN

Penelitian akan menerapkan pendekatan metodologi terstruktur untuk mengimplementasikan JWT dalam proses autentikasi dan otorisasi pada sistem pembayaran digital. Proses pengembangan dilakukan melalui beberapa tahap, meliputi identifikasi masalah, studi literatur, analisis kebutuhan sistem, perancangan arsitektur keamanan, implementasi JWT sebagai komponen utama, dan pengujian kinerja sistem untuk memastikan aspek keamanan dan efisiensi.



Gambar 1. Tahapan metode penelitian

Tahap pertama penelitian ini mencakup identifikasi masalah yang bertujuan untuk mengidentifikasi masalah yang akan dibahas. Fokus penelitian berada pada sistem pembayaran digital yang masih rentan terhadap keamanan arus komunikasi data.

Tahap berikutnya adalah studi literatur yang bertujuan memperoleh pemahaman mengenai prinsip autentikasi, otorisasi, dan keamanan data, termasuk konsep *JSON Web Token* (JWT). Kajian dilakukan dengan mengumpulkan referensi dari jurnal ilmiah dan artikel yang relevan. Literatur yang digunakan mencakup teknologi JWT, penggunaan Python Flask, serta penerapan mekanisme autentikasi dan otorisasi pada sistem.

Setelah pengumpulan data, tahap selanjutnya adalah analisis kebutuhan sistem. Tahap ini mencakup analisis kebutuhan keamanan data, mekanisme autentikasi dan otorisasi, serta skalabilitas sistem. Hasil analisis digunakan untuk menentukan pengembangan sistem pembayaran digital.

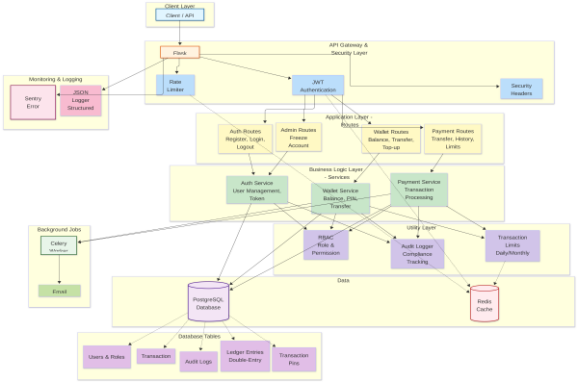
Tahap selanjutnya adalah perancangan sistem autentikasi dan otorisasi menggunakan JWT dan Flask dalam sistem pembayaran digital. Perancangan tersebut mencakup pemodelan struktur token JWT, pemilihan algoritma enkripsi, serta mekanisme validasi token, dengan mempertimbangkan aspek keamanan dan skalabilitas. Proses verifikasi token dilakukan dengan memeriksa keabsahan dan masa berlaku token. Jika token valid, sistem mengevaluasi hak akses yang terkandung dalam *payload* sebelum memberikan respons berupa data atau layanan yang diminta oleh pengguna [13].

Tahap implementasi sistem dalam penelitian ini akan dilakukan dengan mengimplementasikan desain sistem autentikasi dan otorisasi menggunakan JWT pada Flask sebagai alat dan kerangka kerja untuk memvalidasi komunikasi sistem berbasis token dalam pengembangan API. Selain itu, logika penggunaan JWT, manajemen token, proses validasi, dan fitur terkait keamanan lainnya akan diimplementasikan pada sistem backend.

Tahap terakhir adalah menguji sistem menggunakan Postman sebagai alat untuk menguji keamanan API yang menerapkan autentikasi dan otorisasi berbasis JWT. Pengujian dilakukan menggunakan metode *black-box testing*, yaitu pengujian yang berfokus pada perilaku eksternal sistem tanpa memperhatikan struktur internal kode program. Keberhasilan pengujian ditentukan berdasarkan kesesuaian respons HTTP yang diterima dengan hasil yang diharapkan pada setiap skenario uji. Penilaian dilakukan dengan memastikan bahwa sistem menghasilkan kode status HTTP yang tepat, serta memiliki struktur dan isi *body response* yang sesuai dengan spesifikasi yang telah dirancang. Selain itu, sistem juga diuji kemampuannya dalam menolak input yang tidak valid maupun berpotensi berbahaya, serta menjaga konsistensi data sebelum dan sesudah proses transaksi dilakukan. Pengujian dinyatakan berhasil apabila seluruh indikator tersebut terpenuhi sesuai dengan logika bisnis sistem yang telah ditetapkan.

4. HASIL DAN PEMBAHASAN

Analisis kebutuhan sistem pada layanan pembayaran digital bertujuan untuk menjamin keamanan dan performa yang optimal. Kebutuhan fungsional mencakup proses autentikasi pengguna, yaitu sistem harus dapat memverifikasi identitas pengguna melalui email dan kata sandi; pembuatan token, yakni kemampuan sistem untuk menghasilkan token JWT setelah proses autentikasi berhasil dilakukan; validasi token, yang memastikan setiap permintaan pengguna diperiksa melalui verifikasi JWT; otorisasi, yang berfungsi mengelola hak akses berdasarkan peran dan izin yang terenkapsulasi dalam token; serta dukungan terhadap *refresh token* untuk memperbarui *access token* yang telah kedaluwarsa.



Gambar 2. Arsitektur sistem pembayaran digital

Gambar arsitektur sistem menunjukkan desain berlapis (*layered architecture*) yang terdiri dari lapisan utama. Penggunaan *layered architecture* dalam pengembangan perangkat lunak terbukti mampu meningkatkan modularitas sistem dengan membagi aplikasi ke dalam lapisan presentasi, logika bisnis, dan akses data yang memiliki tanggung jawab terpisah secara jelas [14].

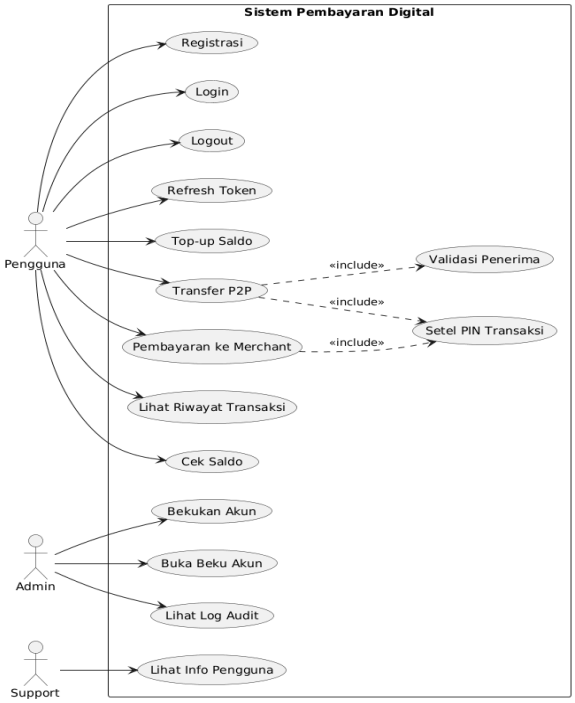
Lapisan terluar dimulai dari klien sebagai konsumen layanan yang mengakses API melalui protokol HTTP. Permintaan dari klien kemudian diteruskan ke lapisan *gateway* dan keamanan yang dikelola oleh aplikasi Flask sebagai server utama. Pada lapisan tersebut, terdapat mekanisme autentikasi JWT untuk verifikasi identitas pengguna tanpa menyimpan sesi di server, komponen *rate limiter* untuk mencegah serangan *denial-of-service.*

Setelah melewati lapisan keamanan, permintaan diteruskan ke lapisan rute aplikasi yang mengelola endpoint API. Rute autentikasi menangani registrasi pengguna baru, login dengan kredensial, dan logout dengan pembatalan token. Rute *wallet* menyediakan endpoint untuk pengecekan saldo, transfer antar pengguna, dan pengisian ulang saldo. Rute payment memfasilitasi transaksi pembayaran, riwayat transaksi, dan pengecekan batas transaksi harian atau bulanan. Rute administrasi memberikan akses khusus bagi administrator untuk melakukan tindakan seperti pembekuan akun pengguna yang melanggar kebijakan. Setiap rute berkomunikasi dengan lapisan logika bisnis yang mengimplementasikan aturan bisnis dan pemrosesan data sesuai prinsip *separation of concerns*.

Lapisan logika bisnis terdiri dari tiga layanan utama yang menangani operasi kritis sistem. Layanan autentikasi mengelola siklus pengguna termasuk pembuatan akun, validasi kredensial, dan pembuatan token akses. Layanan *wallet* bertanggung jawab atas operasi dompet digital seperti pengecekan saldo, validasi PIN, dan eksekusi transfer dana. Layanan pembayaran memproses transaksi dengan menerapkan validasi multitahap, pencatatan ledger ganda (*double-entry bookkeeping*), dan pengecekan batas transaksi. Ketiga layanan tersebut memanfaatkan lapisan utilitas yang menyediakan fungsi pendukung seperti modul RBAC untuk mengelola hak akses berbasis peran, komponen audit logger untuk mencatat aktivitas kritis guna keperluan kepatuhan regulasi, dan modul pemeriksaan batas transaksi untuk memvalidasi setiap transaksi terhadap batas harian dan bulanan yang ditetapkan.

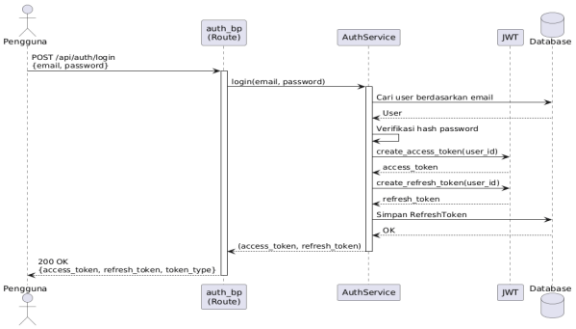
Lapisan data mengelola pengambilan informasi melalui dua komponen utama. Basis data relasional menyimpan data terstruktur dengan dukungan transaksi ACID dan skema yang dirancang dengan normalisasi untuk menghindari redundansi data. Struktur basis data terdiri dari tabel pengguna dan peran untuk menyimpan informasi pengguna beserta kredensial terenkripsi, tabel transaksi untuk mencatat setiap pembayaran dengan status dan metadata terkait, tabel *entri ledger* yang mengimplementasikan sistem pembukuan ganda, tabel *log audit* untuk keperluan kepatuhan dan investigasi, serta tabel PIN transaksi untuk menyimpan PIN terenkripsi. Penyimpanan *cache* Redis digunakan untuk menyimpan data sementara seperti token sesi, dan hasil *query* yang sering diakses, sehingga mengurangi beban basis data utama dan meningkatkan waktu respons sistem.

Berdasarkan arsitektur sistem yang telah dibuat maka dilakukan perancangan sistem yang digambarkan melalui *Unified Modeling Language*, yang meliputi *use case diagram* dan *sequence diagram*.



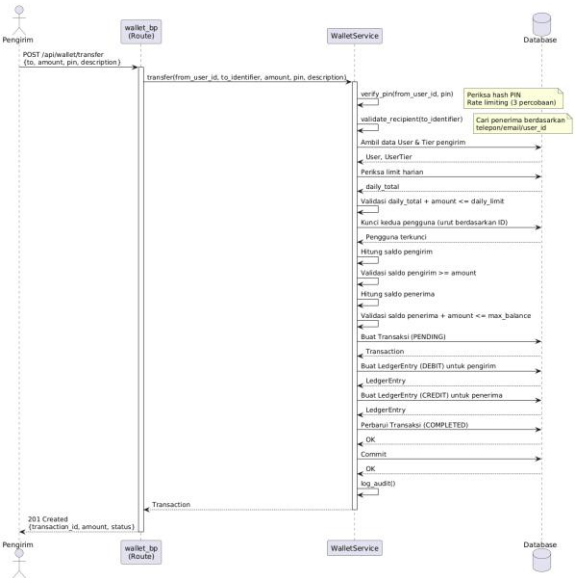
Gambar 3. Use case diagram sistem

Pada Gambar 3 menunjukkan *use case diagram* untuk sistem pembayaran digital yang terdiri atas tiga kelompok fungsional utama, yaitu autentikasi, wallet, dan admin. Aktor yang terlibat meliputi pengguna,admin, dan *support*, yang masing-masing berinteraksi dengan fungsionalitas berbeda sesuai perannya. Pada modul Autentikasi, pengguna dapat melakukan registrasi, *login*, *logout*, serta pembaruan token. Modul Manajemen *Wallet* memuat berbagai proses transaksi seperti *top-up* saldo, transfer P2P, pemeriksaan saldo, dan peninjauan riwayat transaksi, dengan beberapa proses pendukung seperti validasi penerima dan pengaturan PIN transaksi yang direlasikan menggunakan relasi *«include»*. Sementara itu, modul Manajemen Admin memungkinkan admin melakukan pembekuan dan pembukaan pembekuan akun, melihat *log audit*, serta memeriksa informasi pengguna. Diagram sistem pembayaran digital memberikan gambaran menyeluruh mengenai ruang lingkup fungsional dan interaksi antara aktor serta sistem.



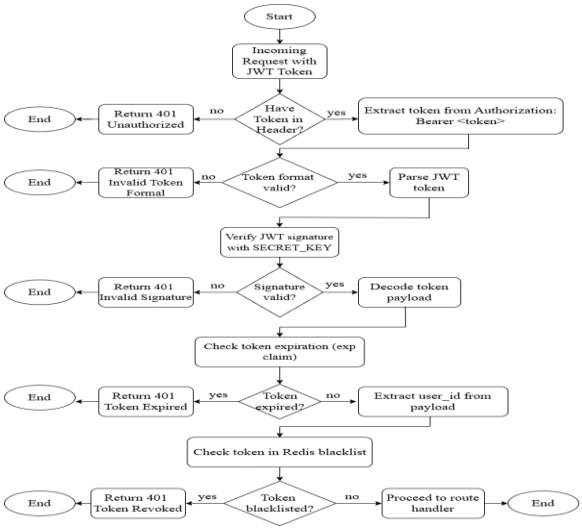
Gambar 4. Sequence diagram login

Gambar 4 menggambarkan proses autentikasi pengguna ketika melakukan *login*. Pengguna mengirim permintaan POST ke rute */api/auth/login* yang kemudian diteruskan ke fungsi *AuthService*. Layanan akan mencari data pengguna berdasarkan email, memverifikasi kecocokan kata sandi menggunakan hash, dan setelah validasi berhasil sistem membuat *access token* serta *refresh token* menggunakan komponen JWT. Token yang dihasilkan disimpan ke dalam database. Setelah seluruh proses selesai, sistem mengembalikan respons berisi *access token*, *refresh token*, dan *token type*.



Gambar 5. *Sequence diagram* transfer

Gambar 5 menjelaskan alur proses transfer antarpengguna pada modul *wallet*. Proses dimulai ketika pengguna mengirim permintaan POST ke rute */api/wallet/transfer* yang diteruskan ke *WalletService*. Sistem melakukan serangkaian validasi, termasuk verifikasi PIN, validasi identitas penerima, pemeriksaan batas harian, penguncian pengguna untuk menghindari *race condition*, serta validasi saldo pada pengirim dan penerima. Setelah seluruh kondisi terpenuhi, sistem membuat transaksi berstatus *pending*, mencatat *ledger entry* untuk debit dan kredit, kemudian memperbarui status transaksi menjadi *completed* sebelum menyimpan *log audit*.



Gambar 6. Alur verifikasi JWT

Proses verifikasi JWT diawali saat *frontend* mengajukan permintaan ke *backend* dengan menyertakan token, yang kemudian diverifikasi melalui pemeriksaan *signature* dan masa berlakunya. Apabila token sah, sistem mengekstrak hak akses dari *payload* dan memberikan respons data sesuai izin pengguna. Sebaliknya, jika token tidak sah atau telah kedaluwarsa, sistem menolak permintaan dengan menampilkan pesan kesalahan dalam bentuk JSON [15].

Rancangan layanan disusun secara modular untuk mempermudah integrasi dengan Python menggunakan framework Flask, dengan JWT memanfaatkan algoritma HMAC SHA-256 dan pengujian validitas token dilakukan melalui Postman. Tujuan utama dari penerapan JWT adalah untuk meningkatkan aspek keamanan dalam proses pertukaran data antarsistem dengan cara menyertakan token pada setiap *request* yang dikirim oleh klien [16].

Pengujian sistem pembayaran digital berbasis JWT memerlukan pendekatan komprehensif untuk memastikan keamanan, keandalan, dan kinerja sistem dalam berbagai kondisi. Serangkaian pengujian yang dilakukan dalam penelitian mencakup empat aspek, yaitu pengujian keamanan JWT, pengujian keamanan API, pengujian fungsionalitas alur transaksi, dan pengujian *concurrency*.

Tabel 1. Hasil pengujian keamanan JWT

Kasus Uji	**Metode**	**Hasil yang diharapkan**	**Status Uji**
Validasi kedaluwarsa token	Menggunakan token kedaluwarsa	HTTP 401	Lulus
Deteksi *signature* tidak valid	Memanipulasi *signature* token	HTTP 401	Lulus
Algoritma *none*	Membuat token dengan alg = "*none*" tanpa *signature*	HTTP 401	Lulus

Kasus Uji	Metode	Hasil yang diharapkan	Status Uji
Tanpa *exp* di *claims* pada token	Menggunakan token tanpa *expiration*	HTTP 401	Lulus

Tabel 1 Pengujian Keamanan JWT menyajikan serangkaian uji untuk memverifikasi mekanisme keamanan token yang digunakan dalam autentikasi sistem. Dengan penggunaan JWT, terutama dengan algoritma kriptografis yang kuat seperti HMAC-SHA512, dapat memperkuat keamanan sesi dan akses API dengan meminimalkan celah autentikasi seperti *replay attack* dan manipulasi token [13].

Kasus uji pertama mengenai token kedaluwarsa memastikan bahwa sistem menolak token yang sudah tidak berlaku, yang sejalan dengan praktik keamanan terbaik JWT. Hal ini penting karena *claim exp* pada *JSON Web Token* (JWT) berperan dalam menentukan masa berlaku token sehingga dapat membantu mencegah potensi *session hijacking*. Uji selanjutnya tentang *signature* menegaskan bahwa *signature* bertindak sebagai bukti integritas token. Selain itu, pengujian algoritma *none* dan klaim *exp* tanpa *expiration* menegaskan bahwa server tidak mengabaikan parameter keamanan krusial JWT. Oleh karena itu, mekanisme validasi JWT dalam penelitian terbukti konsisten dengan teori bahwa sistem yang memanfaatkan klaim terstruktur dan verifikasi kriptografi dapat mengurangi peluang penyalahgunaan token dan akses ilegal.

Hasil pengujian *concurrency* pada Tabel 2 menunjukkan bahwa sistem berhasil menjaga konsistensi data secara penuh dengan tingkat keberhasilan 100% pada seluruh skenario yang diuji. Dari 10 permintaan *top-up* bersamaan dengan saldo awal Rp0, sistem menghasilkan saldo akhir yang valid sebesar Rp500.000 berkat mekanisme *database locking* yang memastikan setiap proses baca dan tulis data saldo dilakukan secara atomik. Sistem menangani *race condition* melalui penerapan transaksi database ACID, yang menjamin setiap permintaan diproses secara terisolasi tanpa interferensi antar permintaan sehingga inkonsistensi data dapat sepenuhnya dicegah. Selain itu, mekanisme *idempotency key* diimplementasikan melalui header *X-Idempotency-Key* yang wajib disertakan pada setiap permintaan transaksi, di mana sistem menyimpan *key* tersebut di Redis sehingga permintaan duplikat dengan *key* yang sama akan langsung mengembalikan respons dari permintaan pertama tanpa memproses ulang transaksi terbukti dari skenario 10 permintaan dengan *key* identik yang hanya menghasilkan satu transaksi berhasil dan saldo bertambah Rp100.000. Pengujian lebih lanjut mengonfirmasi bahwa sistem menolak permintaan yang melampaui *daily limit* Rp2.000.000, serta mekanisme proteksi *brute force* pada autentikasi PIN berjalan dengan benar ketika PIN terkunci setelah tiga kali percobaan yang salah secara bersamaan.

Pengujian fungsionalitas alur transaksi pada Tabel 3 dilakukan untuk menilai kesesuaian setiap *endpoint* dengan logika bisnis yang dirancang. Pengujian pengaturan PIN enam digit memastikan sistem menerapkan fitur keamanan tambahan sesuai standar perlindungan pengguna finansial, sementara pengecekan saldo sebelum dan sesudah *top-up* membuktikan pembaruan saldo yang akurat pada setiap transaksi. *Top-up* valid menghasilkan respons HTTP 201, sedangkan transaksi dengan jumlah negatif dan permintaan kepada penerima yang tidak ditemukan ditolak sistem guna mencegah inkonsistensi data dan perolehan saldo yang tidak sah.

Tabel 2. Hasil pengujian concurrency

Skenario Pengujian	Saldo Awal	Jumlah *Request* Bersamaan	Jumlah Uang per *Request*	Hasil yang diharapkan	Status
POST */api/wallet/topup* 10 *concurrent top-up* dengan *idempotency key* berbeda	Rp 0	10	Rp 50.000	HTTP 201 - Saldo akhir adalah Rp 500.000.	Lulus
POST */api/wallet/topup* 10 *concurrent top-up* dengan *idempotency key* sama	Rp 0	10	Rp 100.000	Satu HTTP 201 dan sembilan HTTP 400. Saldo bertambah sebesar Rp 100.000.	Lulus
POST */api/wallet/transfer* *Concurrent* transfer melebihi daily limit	Rp 5.000.000	10	Rp 500.000	Empat HTTP 201 dan enam HTTP 400 hingga batas harian sebesar Rp 2.000.000 tercapai	Lulus
POST */api/wallet/pin* *Concurrent* transfer dengan PIN salah	Rp 3.000.000	3	Rp 100.000	HTTP 400 – PIN terkunci setelah tiga kali percobaan gagal	Lulus

Tabel 3. Hasil pengujian fungsionalitas alur transaksi

Skenario Pengujian	*Endpoint*	Hasil yang diharapkan	Status
Mengatur PIN transaksi 6 digit	POST */api/wallet/pin*	HTTP 200 - PIN berhasil diatur	Lulus
Cek saldo awal	GET */api/wallet/balance*	HTTP 200 - Menampilkan saldo	Lulus
Top-up saldo Rp 500.000	POST */api/wallet/topup*	HTTP 201 - Topup berhasil	Lulus
Validasi penerima transfer yang tidak ada	GET */api/wallet/validate-recipient*	HTTP 404 - Penerima tidak ditemukan	Lulus
Transfer melebihi saldo	POST */api/wallet/topup*	HTTP 400 - Saldo tidak mencukupi	Lulus
Ambil riwayat transaksi	GET */api/wallet/history*	HTTP 200 - Menampilkan list transaksi	Lulus
Top-up dengan amount negatif	POST */api/wallet/topup*	HTTP 400 - Jumlah harus positif	Lulus
Akses saldo setelah logout	GET */api/wallet/balance*	HTTP 401 - *Token revoked*	Lulus

Tabel 4. Hasil pengujian keamanan API

Jenis Kerentanan	*Endpoint*	*Request* yang diuji	Hasil yang diharapkan	Status
Broken Authentication	POST */api/auth/login*	Melakukan serangan *brute force* 1000 *request*	Memberikan respons HTTP 429 ketika melebihi batas maksimal *request* per menit	Lulus
Broken Access Control (Privilege Escalation)	GET */api/admin/audit-logs*	Pengguna biasa melakukan *request* terhadap *endpoint* yang membutuhkan role admin	HTTP 403 *FORBIDDEN*	Lulus
Sensitive Data Exposure	GET */api/auth/profile*	Melihat respons yang diberikan	Respons tidak memberikan data sensitif seperti kata sandi.	Lulus
Rate Limiting Absence	POST */api/wallet/transfer*	Melakukan 500 *request* per detik	Memberikan respons HTTP 429 ketika melebihi batas maksimal *request* per detik	Lulus
Missing Security Headers	Semua *endpoint*	Melakukan *request* terhadap semua *endpoint* yang tersedia	Menampilkan pada respons *header* ● X-Content-Type-Options ● X-Frame-Options, 7.6 cm ● X-XSS-Protection ● Strict-Transport-Security (HSTS)	Lulus
SQL Injection	POST */api/wallet/transfer*	to: “OR ‘1’=’1”	HTTP 400 *BAD REQUEST*	Lulus

Pengujian keamanan API pada Tabel 4 mengacu pada standar OWASP API Security dan menunjukkan hasil yang komprehensif pada seluruh kategori kerentanan. Sistem merespons percobaan *brute force* dengan HTTP 429 sebagai indikasi perlindungan terhadap serangan massal pada *endpoint* login, menolak akses pengguna non-admin pada *endpoint* khusus admin, serta memastikan data sensitif seperti kata sandi tidak bocor melalui respons API. *Rate Limiting* berjalan efektif dan *SQL Injection* berhasil ditolak, membuktikan validasi input telah diterapkan dengan benar. Hal ini sejalan dengan prinsip bahwa arsitektur API harus mengombinasikan kontrol autentikasi seperti JWT dengan mekanisme keamanan lain seperti *rate limiting* dan validasi input untuk mencegah eksploitasi [17].

Secara keseluruhan, rangkaian pengujian yang telah dilakukan, meliputi pengujian keamanan JWT, pengujian keamanan API, pengujian fungsionalitas alur transaksi, serta pengujian *concurrency* menunjukkan bahwa sistem mampu beroperasi secara konsisten dalam berbagai skenario penggunaan.

5. KESIMPULAN DAN SARAN

Hasil dari penelitian mengindikasikan bahwa penerapan *JSON Web Token* dengan memanfaatkan Flask mampu meningkatkan keamanan pada sistem pembayaran digital, terutama dalam hal autentikasi dan otorisasi API. Pendekatan tersebut memungkinkan penerapan kontrol akses yang lebih sistematis, sehingga dapat meminimalisir penyalahgunaan hak akses yang sering menjadi tantangan dalam sistem pembayaran digital. Berdasarkan seluruh pengujian yang dilakukan, sistem menunjukkan hasil yang sangat baik pada aspek keamanan, fungsionalitas, maupun konsistensi data. Pengujian keamanan JWT berhasil memvalidasi berbagai skenario, seperti token kedaluwarsa, *signature* tidak valid, penggunaan algoritma *none*, serta token tanpa klaim *exp*, yang seluruhnya ditolak oleh sistem dengan respons HTTP 401 sesuai standar keamanan. Selain itu, pengujian keamanan API berdasarkan standar OWASP juga menunjukkan bahwa sistem mampu menangani berbagai potensi kerentanan dengan seluruh pengujian dinyatakan lulus. Pada sisi fungsionalitas, pengujian alur transaksi yang mencakup skenario pengaturan PIN, *top-up* saldo, transfer dana, validasi penerima, hingga riwayat transaksi berhasil dijalankan dengan baik dan menghasilkan kode status HTTP yang sesuai pada setiap proses.

Sementara itu, pengujian *concurrency* menunjukkan bahwa sistem tetap mampu menjaga konsistensi saldo meskipun menerima hingga 10 permintaan secara bersamaan. Dengan penggunaan

token yang menggunakan algoritma HMAC SHA-256, JWT memberikan kejelasan mengenai hak akses pengguna, yang secara efektif menutup celah keamanan yang rentan terhadap eksploitasi oleh pihak yang tidak berwenang. Penelitian yang dilakukan masih memiliki ruang untuk pengembangan lebih lanjut, dengan mengintegrasikan JWT bersama teknologi keamanan seperti *Multi Factor Authentication* guna memperkuat lapisan perlindungan.

DAFTAR PUSTAKA

[1] A. R. Maulidah, R. P. Astuti, K. Nisa, W. Erlangga, and E. Hambarwati, "Perkembangan Sistem Pembayaran Digital : Pada Era Revolusi Industri 4.0 Di Indonesia," *Jurnal Ekonomi Dan Bisnis Digital*, vol. 01, no. 04, pp. 798–803, Jun. 2024.

[2] S. A. Sudari, "Implementation of Financial Technology 'Digital Payment' on Consumer Behavior at SMEs Depok City in Digital Transformation Era," *Jurnal Administrasi Karya Dharma*, vol. 3, no. 1, p. 2024, 2024.

[3] H. Humairoh, M. Annas, A. Wiwaha, T. Duha, and E. Endri, "Digitalization of the payment systems: evidence from Indonesia," *Quality - Access to Success*, vol. 26, no. 205, pp. 245–254, 2025, doi: 10.47750/QAS/26.205.25.

[4] N. Wulandari, A. Wibowo, and B. Susanto, "Penerapan RESTful API untuk Membangun Program Pembayaran Piutang Menggunakan Otentikasi OAuth 2.0," *Jurnal Terapan Teknologi Informasi*, vol. 5, no. 1, pp. 1–10, Apr. 2021, doi: 10.21460/jutei.2021.51.230.

[5] B. C. Utomo and A. A. Rahman, "Analisis Kesadaran Keamanan Data Pribadi pada Pengguna E-Wallet DANA," *JRST (Jurnal Riset Sains dan Teknologi)*, vol. 8, no. 2, p. 155, Oct. 2024, doi: 10.30595/jrst.v8i2.21162.

[6] Y. Susila Atmaja and D. Hartono Paulus, "Partisipasi Bank Indonesia Dalam Pengaturan Digitalisasi Sistem Pembayaran Indonesia," *Jurnal Masalah-Masalah Hukum*, vol. 51, no. 3, Jul. 2022.

[7] I. Melyana, M. S. Darmawan, S. Syukur, M. I. A. Syah, and F. Purwani, "Algoritma Autentikasi Multi Factor Sebagai Solusi Untuk Meningkatkan Keamanan Sistem Pembayaran E-Commerce," *Digital Transformation Technology*, vol. 4, no. 2, pp. 984–989, Dec. 2024, doi: 10.47709/digitech.v4i2.4969.

[8] A. Y. Nashikhuddin, J. Karaman, and Y. Litanianda, "Implementasi Api Restful Dengan *Json Web Token* (JWT) Pada Aplikasi E-Commerce Thrifty Shop Untuk Otentikasi Dan Otorisasi Pengguna," *METHOMIKA: Jurnal Manajemen Informatika & Komputerisasi Akuntansi*, vol. 7, no. 2, 2023, doi: 10.46880/jmika.Vol7No2.pp239-246.

[9] I. Darmawan, M. Umar Mansyur, K. Zulfana Imam, and M. Syahdan, "Evaluasi Keamanan Privilege Terintegerasi *JSON Web Token* pada Sistem Informasi Akademik," *Jurnal Informasi dan Teknologi*, vol. 5, Jul. 2023, doi: 10.37034/jidt.v5i1.368.

[10] S. Dalimunthe, E. Hasri Putra, and M. A. Fadhly Ridha, "Restful API Security Using *JSON Web Token* (JWT) With HMAC-Sha512 Algorithm in Session Management," *IT Journal Research and Development*, vol. 8, no. 1, pp. 81–94, Dec. 2023, doi: 10.25299/itjrd.2023.12029.

[11] M. D. Anwar and I. A. Kautsar, "Arsitektur Perangkat Lunak Berbasis Layanan Mikro pada Sistem Manajemen Informasi Kantin," *Physical Sciences, Life Science and Engineering*, vol. 1, no. 2, p. 13, Jan. 2024, doi: 10.47134/pslse.v1i2.196.

[12] R. Nandang Pratama and Y. A. Susetyo, "Implementasi Python API dengan Framework Flask sebagai Cloud Run Service Untuk Proses Update di PT. XYZ," *KESATRIA: Jurnal Penerapan Sistem Informasi (Komputer & Manajemen)*, vol. 5, no. 2, pp. 669–676, Apr. 2024, doi: https://doi.org/10.30645/kesatria.v5i2.376.

[13] Y. Loui Pattinama, Ferdyansyah, I. Susanti, and Painem, "Implementasi Rest API Web Service Dengan Otentifikasi *JSON Web Token* Untuk Aplikasi Properti," *Jurnal Informatik*, no. 1, Apr. 2023.

[14] Z. Tu, "Research on the Application of Layered Architecture in Computer Software Development," *Journal of Computing and Electronic Information Management*, vol. 11, no. 3, p. 34, 2023.

[15] E. R. Setiawan, A. Fajaryanto Cobantoro, and M. B. Setyawan, "Implementasi Authentication & Authorization Berbasis JWT Pada Sistem Pengelolaan Perkuliahan Menggunakan Algoritma Hmac," *Multitek Indonesia: Jurnal Ilmiah*, vol. 19, pp. 1–16, Jul. 2025, [Online]. Available: http://journal.umpo.ac.id/index.php/multitek

[16] A. Ramadhan and S. Mulyati, "Implementation Of *Json Web Token* In The Development Of Village Monograph Database Based On Restapi," *Jurnal Teknik Informatika (Jutif)*, vol. 5, no. 6, pp. 1849–1860, Dec. 2024, doi: 10.52436/1.jutif.2024.5.6.4028.

[17] Shivam and M. Gupta, "Secure API Gateway with Rate Limiting and JWT Authentication," *Int. J. Res. Appl. Sci. Eng. Technol.*, vol. 13, no. 4, pp. 3559–3562, Apr. 2025, doi: 10.22214/ijraset.2025.68953.