Progresif: Jurnal Ilmiah Komputer
Jl. Ahmad Yani, K.M. 33,5 - Kampus STMIK Banjarbaru
Loktabat – Banjarbaru (Tlp. 0511 4782881), e-mail: puslit.stmikbjb@gmail.com
e-ISSN: 2685-0877
p-ISSN: 0216-3284

Implementasi Sistem Autentikasi JSON *Web* Token Pada Aplikasi *Fieldrent* Menggunakan Algoritme SHA-512

Brian Marcius Ega Wijaya[1*], Felix Andreas Sutanto[2]
Program Studi Teknik Informatika, Universitas Stikubank Semarang, Semarang, Indonesia
*e-mail *Corresponding Author*: brianmarcius27@gmail.com

Abstract

The times and the Covid-19 pandemic have driven the shift to digital technology and online activities, including in the field rental sector. In addition to encouraging technological advances, the pandemic has also had a major impact on global economic problems and increased crime, including cyber crime. So it is important to develop an application security system with user authentication processes and sensitive data encryption to prevent the spread of data to irresponsible parties. Implementing Json Web Token with SHA-512 as an authentication process when accessing the restful API is an effort to avoid cyber crime in the Fieldrent application. The SHA-512 algorithm has high complexity so it will not be easily solved using Brute-force and other cryptanalysis attacks. The results of this study indicate that the implementation of an authentication system using the JWT token with the SHA-512 algorithm is safer in securing API access than conventional authentication systems.

Keywords*: Authentication, Cryptography, SHA-512, Eeb security, JWT token*

Abstrak

Perkembangan zaman dan pandemi *Covid-19* mendorong peralihan ke teknologi digital dan aktivitas secara *online* termasuk pada sektor persewaan lapangan. Selain mendorong kemajuan teknologi pandemi juga menyebabkan dampak yang besar dalam masalah ekonomi global dan meningkatnya kejahatan termasuk cyber crime. Sehingga penting untuk mengembangkan sistem keamanan aplikasi dengan proses autentikasi pengguna dan enkripsi data sensitif untuk mencegah penyebaran data kepada pihak yang tidak bertanggung jawab. Mengimplementasikan *Json Web* Token dengan SHA-512 sebagai proses autentikasi saat mengakses restful API menjadi upaya menghindari cyber crime pada aplikasi Fieldrent. Dengan Algoritme SHA-512 memiliki kompleksitas yang tinggi sehingga tidak akan mudah dipecahkan menggunakan Brute-force dan serangan kriptanalisis lainnya. Hasil penelitian ini menunjukan bahwa Implementasi sistem autentikasi menggunakan JWT token dengan Algoritme SHA-512 lebih aman dalam mengamankan akses API dibandingkan sistem autentikasi konvensional.

Kata kunci*: Autentikasi; Kriptografi; SHA-512; Web security; JWT token*

1. Pendahuluan

Teknologi Informasi dan komunikasi telah berkembang pesat beberapa tahun terakhir. Perkembangannya bisa dilihat dari banyaknya aplikasi baik yang berbasis web maupun berbasis mobile. Banyak sektor yang terdampak dalam perkembangan teknologi ini salah satunya yaitu sektor sewa lapangan. Dibalik perkembangan teknologi yang sangat cepat ini juga diikuti dengan perkembangan *cyber crime* yang semakin tinggi. *Cyber crime* atau dalam bahasa Indonesia disebut dengan kejahatan dunia maya adalah pelanggaran yang hanya dapat dilakukan menggunakan komputer, jaringan komputer atau bentuk lain dari teknologi komunikasi informasi [1]. Pencurian data-data sensitif dan disebarluaskan untuk kepentingan pribadi merupakan salah satu contoh *cyber crime* yang belakangan ini terjadi.

Menurut laporan terkini dari Indeks Keamanan Siber Nasional (NCSI), Indonesia ditempatkan pada peringkat 84 dengan skor 38,96 dalam hal keamanan siber [2]. Dalam laporan tersebut, NCSI menggunakan 12 indikator yang meliputi berbagai aspek, mulai dari

perkembangan kebijakan keamanan siber, perlindungan data pribadi, hingga upaya dalam melawan kejahatan siber [2]. Laporan NCSI tersebut menunjukkan bahwa tingkat keamanan siber di Indonesia masih tergolong rendah dibandingkan negara-negara lain.

Faktor lain yang menyebabkan rendahnya tingkat keamanan siber di Indonesia adalah lemahnya sistem keamanan dalam aplikasi yang menyebabkan bocornya data ke pihak yang tidak bertanggung jawab *(sensitive data exposure)*. *Sensitive data* meliputi informasi yang perlu dijaga keamanannya dari serangan, seperti kata sandi, alamat, nomor keamanan sosial, kartu kredit, informasi perbankan, catatan medis [3]. Di sisi lain, *data exposure* merujuk pada situasi di mana data atau informasi tidak terlindungi dengan baik atau terpapar sehingga pelaku serangan memiliki kesempatan untuk mengeksploitasi dan mencuri data [3].

Sistem keamanan pada aplikasi sangat diperlukan dalam mencegah adanya kebocoran data pada aplikasi. Sistem autentikasi konvensional yang sering digunakan hanya membatasi pengguna untuk masuk ke dalam aplikasi menggunakan *username* dan *password* atau biasa disebut dengan *single factor authentication* tidak cukup membatasi pengguna untuk mengakses data langsung ke API. Diperlukanlah sebuah kriptografi untuk menyandi data agar data tidak mudah untuk dibaca oleh orang yang tidak bertanggung jawab.

Kriptografi adalah sebuah disiplin ilmu atau seni yang bertujuan untuk mempertahankan keamanan pesan saat dikirimkan dari satu lokasi ke lokasi lainnya.. Asal usul kata kriptografi berasal dari bahasa Yunani dengan kata dasar *crypto* yang berarti rahasia atau *secret* dalam Bahasa Indonesia, dan kata *graphia* yang berarti tulisan atau *writing* [4]. JSON Web Token (JWT) adalah sebuah objek JSON yang telah dienkripsi dan digunakan untuk mentransfer data antar platform. JWT juga bisa digunakan sebagai mekanisme autentikasi saat mengakses REST API [5]. Representational State Transfer (REST) *Application Programming Interface* (API) adalah sebuah antarmuka pemrograman yang mengikuti standar arsitektur komunikasi. REST sering digunakan dalam pengembangan situs web dan layanan berbasis aplikasi. Sementara itu, API merupakan sebuah tautan yang memungkinkan interaksi dan pertukaran data antara aplikasi [6]. Dalam penelitian ini penulis melakukan implementasi JWT token kedalam proses autentikasi aplikasi Fieldrent dan akan dikombinasikan dengan Algoritme kriptografi SHA-512 untuk meningkatkan sisi keamanan dari aplikasi tersebut. SHA-512 *(Secure Hash Algorithm 512 bit)* adalah salah satu jenis kriptografi yang digunakan untuk menghasilkan nilai hash yang unik dari suatu data dengan ukuran tetap yaitu 512 bit. Keunikan dari nilai hash ini berarti tidak memungkinkan untuk melakukan rekonstruksi data asli dari nilai hash tersebut dan menjadikannya lebih tahan dari serangan brute-force [7]. *Brute force* adalah sebuah Algoritme untuk memecahkan kode yang akan mencoba semua kemungkinan kombinasi karakter dalam kode tertentu dengan masukan karakter yang panjang untuk mencari solusi [4].

2. Tinjauan Pustaka

Penelitian yang dilakukan oleh Ficry Cahya Ramdani, Alam rahmatulloh, Rahmi Nur Shofa tentang “Implementasi JSON Web Token pada Authentication dengan Algoritme HMAC SHA-256” pada tahun 2022 menjelaskan tentang penerapan sistem autentikasi JWT token pada restful API aplikasi Tim bebersih masjid dengan *framework codeigniter*. Pada penelitian tersebut JWT token dengan Algoritme SHA-256 menghasilkan kecepatan maksimal pada windows server 2019 [8].

Penelitian yang dilakukan oleh Arief Umarjati, Arief Wibowo berjudul “Implementasi JWT pada Aplikasi Presensi dengan Validasi *Fingerprint*, *Abstract Geotagging* dan *Device Checker*” (2020), menjelaskan bahwa JWT token dapat diimplementasikan sebagai sistem autentikasi API pada aplikasi mobile dan dikombinasikan dengan sistem validasi lainnya. Penerapan JWT token pada aplikasi presensi terbukti meningkatkan keamanan data pada aplikasi [5].

Penelitian oleh Andri Warda Pratama Putra, Adhitya Bhawiyuga, Mahendra Data, “Implementasi Autentikasi JSON Web Token (JWT) Sebagai Mekanisme Autentikasi Protokol MQTT Pada Perangkat NodeMCU” (2018), yang melakukan penerapan json web token sebagai autentikasi pada aplikasi iot. Hasil pengujian peneliti JWT token dapat diimplementasikan ke dalam perangkat iot. Ketika publisher token yang tidak valid, server dan broker dapat meng-autentikasi. Ketika token yang digunakan telah expired, broker dan server berhasil melakukan autentikasi dan menampilkan pesan error. Ketika hal ini terjadi maka publisher harus melakukan request ulang token. [9].

Penelitian yang dilakukan Andi Setiawan dan Ade Irma Purnamasari, “Implementasi JSON Web Token Berbasis Algoritme SHA-512 untuk Otentikasi Aplikasi Batik Kita” (2020), peneliti melakukan penerapan json web token dengan SHA-512 pada autentikasi sebuah aplikasi bernama Batik Kita dikomparasi dengan SHA-256 dan SHA-384. Pada penelitian tersebut menghasilkan kesimpulan implementasi JSON Web Token (JWT) dengan Algoritme SHA-512 pada aplikasi Batik Kita dapat mempercepat pada saat proses otentikasi serta dapat meningkatkan keamanan karena penggunaan token pada saat otentikasi. Sehingga implementasi JSON Web Token(JWT) dengan Algoritme SHA-512 pada aplikasi Batik Kita berbasis android dan web framework laravel sangat tepat diimplementasikan [10].

Berdasarkan hasil referensi yang dipaparkan di atas dapat disimpulkan bahwa implementasi JWT token pada sistem autentikasi terbukti efektif dalam menangani masalah keamanan API. Aplikasi Fieldrent merupakan aplikasi berbasis web yang memiliki fungsi utama untuk mengelola jadwal sewa lapangan. Aplikasi ini dibangun dengan php native dan menggunakan sistem autentikasi konvensional. Dengan menerapkan sistem autentikasi JWT token dengan Algoritme SHA-512 pada penelitian ini diharapkan dapat meningkatkan keamanan API terutama dalam mencegah kebocoran data sensitif (*sensitive data exposure)*.

3. Metodologi

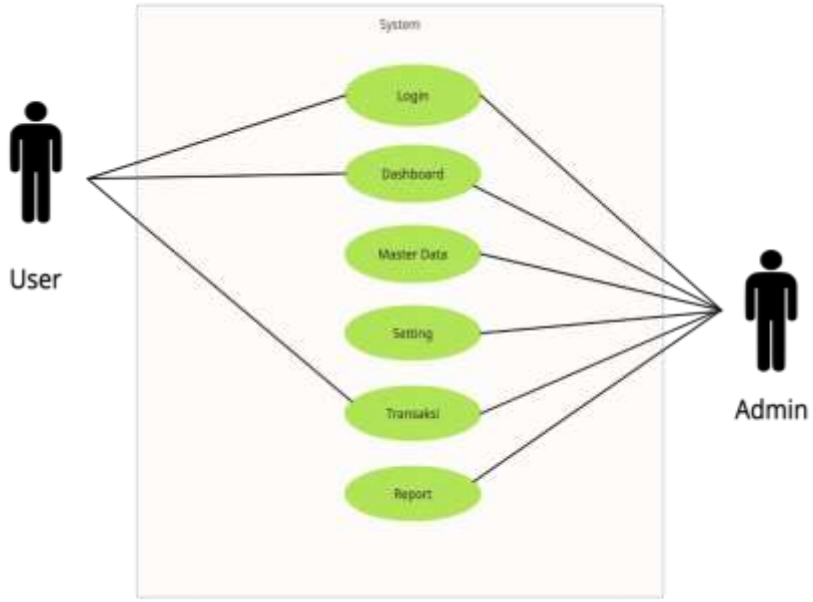
3.1 Analisa Aplikasi

Analisa aplikasi merupakan proses analisa aplikasi untuk mengetahui bagaimana aplikasi tersebut bekerja dan bagaimana bagaimana aturan pengkodean aplikasi tersebut dibuat. Penulis melakukan analisa dengan cara membaca kode projek aplikasi Fieldrent dan mencoba mengaplikasikan field dari login hingga mengakses data.

Aplikasi Fieldrent merupakan aplikasi berbasis web yang digunakan untuk mengelola persewaan lapangan futsal. Aplikasi Fieldrent di install pada web server lokal pada komputer yang disediakan oleh pengguna. Aplikasi Fieldrent dapat diakses melalui alamat http:\\localhost\fieldrent.

Aplikasi Fieldrent memiliki 2 level user yang memiliki akses menu yang berbeda, yaitu :

1. Administrator yang merupakan *superuser* dari aplikasi bertanggung jawab untuk mengelola semua master data (*customers, fields, users*), pengaturan *(payment method)*, transaksi dan juga laporan.
2. *User* bertanggung jawab untuk membantu administrator dalam menginput data transaksi.



Gambar 1. Use Case Diagram aplikasi Fieldrent

Selain dari sisi antarmuka penulis juga melakukan analisa API dengan melakukan akses langsung ke alamat API pada setiap halaman menggunakan postman. Postman adalah sebuah alat yang berguna bagi pengembang yang fokus pada pembuatan API. Fungsi utama dari

Postman adalah sebagai antarmuka pengguna grafis (GUI) untuk pemanggilan API. Namun, saat ini Postman juga menawarkan fitur tambahan, seperti berbagi koleksi API untuk dokumentasi secara gratis, pengujian API secara gratis, kolaborasi tim secara real-time dengan biaya, pemantauan API dengan biaya, dan integrasi dengan biaya [11].

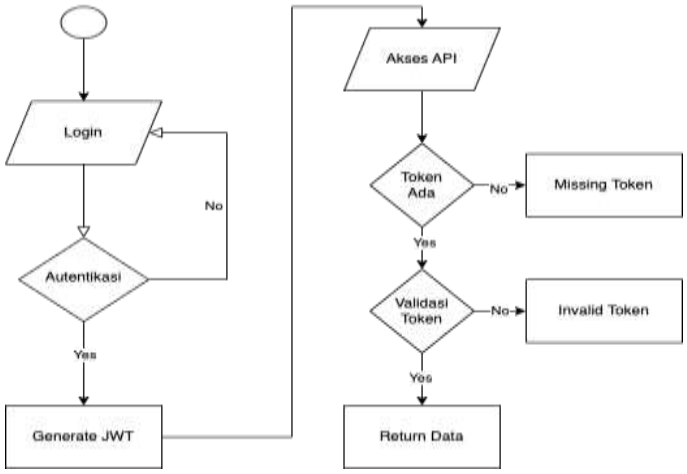
Dalam analisa API yang penulis lakukan menggunakan postman didapatkan hasil seperti yang jelaskan pada tabel 1.

Tabel 1. Tabel Analisa aplikasi Fieldrent

Halaman	***Expectation***	***Current***
Login	Memberikan bukti kredensial berupa token	Tidak memberikan token melainkan dialihkan ke halaman *dashboard*
Customers	Terdapat autentikasi token ketika mengakses API pada menu *customer*	Dapat mengakses API *customer* tanpa ada autentikasi
Fields	Terdapat autentikasi token ketika mengakses API pada menu *fields*	Dapat Mengakses API *fields* tanpa ada proses autentikasi
Settings	Terdapat proses autentikasi token ketika mengakses API pada menu *settings*	Dapat Mengakses API *settings* tanpa ada proses autentikasi
Transactions	Terdapat proses autentikasi token ketika mengakses API pada menu *transactions*	Dapat Mengakses API *transactions* tanpa ada proses autentikasi

Dalam tabel 1 dapat disimpulkan bahwa aplikasi Fieldrent memiliki kelemahan dari segi keamanan akses data *(sensitive data exposure)*, sehingga data yang seharusnya tidak dapat diakses semua orang dapat diakses melalui API tanpa memerlukan autentikasi lebih lanjut.

3.2 Implementasi JWT Token



Gambar 2. Implementasi JWT

Implementasi JWT token sebagai sistem autentikasi pada sebuah aplikasi berbasis web sangat penting terutama untuk web yang sudah menerapkan *Rest API*. Implementasi pengamanan API perlu dilakukan agar pengguna yang tidak bertanggung jawab tidak bisa mengakses langsung ke alamat API tanpa menggunakan bukti kredensial yang valid. Dalam penelitian ini penulis menerapkan JWT dengan alur seperti yang ditunjukkan pada gambar 2.

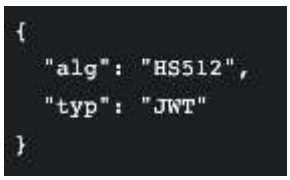
Pada penelitian ini untuk mendapatkan JWT token pengguna aplikasi Fieldrent harus mengakses halaman login dan memasukan email dan kata sandi yang sesuai. Sistem akan mencocokan email dan *password* yang diberikan oleh pengguna dengan data pengguna yang ada di database. Apabila email dan kata sandi yang diberikan salah maka pengguna akan dikembalikan lagi ke halaman login. Sebaliknya apabila email dan kata sandi sesuai maka secara otomatis sistem akan memproses pembentukan JWT. Saat pengguna akan melakukan *request API* maka sistem akan melakukan pengecekan terhadap token yang dikirim oleh pengguna sebagai syarat kredensial pengguna.

3.2.1 Pembentukan JWT

JWT token memiliki 3 komponen utama yang dipisahkan dengan tanda ".", komponen tersebut yaitu *header*, *payload* dan *signature*. Ketiga komponen tersebut dienkripsi kemudian disatukan dengan tanda baca titik[12].

a. Header

Header merupakan bagian pertama dalam JWT, berisi json tipe token dan Algoritme penandatanganan yang digunakan, seperti HMAC SHA512 atau RSA [11]. Pada penelitian kali ini peneliti menggunakan SHA512. Contoh dari *Header* JWT adalah sebagai yang ditunjukkan pada gambar 3.

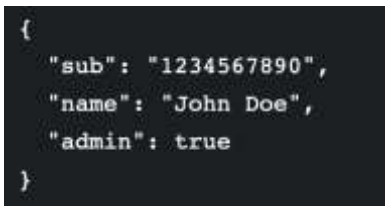


Gambar 3. Contoh json header

Json tersebut akan di enkripsi menggunakan *Base64* untuk membentuk bagian pertama dari JWT.

b. Payload

Bagian kedua dari jwt token adalah *payload*, yang berisikan tentang klaim. Klaim adalah pernyataan tentang entitas biasanya berisi tentang informasi pengguna dan data tambahan lainnya[9]. Ada tiga jenis klaim: klaim terdaftar, publik, dan pribadi. Contoh dari payload sebagaimana yang ditunjukkan pada gambar 4.



Gambar 4. Contoh json payload

Json *payload* tersebut kemudian akan di enkripsi menggunakan *Base64Url* untuk membentuk bagian kedua dari JWT.

c. Signature

Signature merupakan hasil penyandian SHA-512 dari *header*, *payload* dan *secret key. Signature* digunakan untuk memverifikasi bahwa data tidak diubah di sepanjang jalan, dan dalam

kasus token yang ditandatangani dengan kunci pribadi, itu juga dapat memverifikasi bahwa pengirim JWT adalah yang dikatakannya [13].

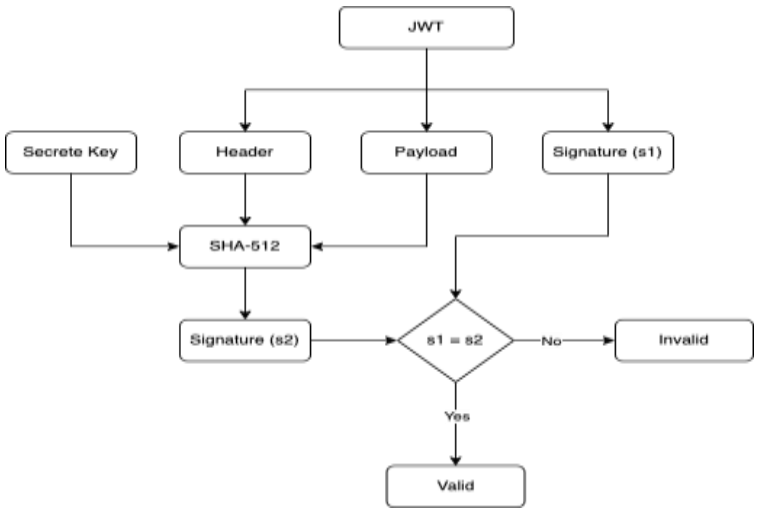
Di dalam fungsi pembentukan JWT, *header* dan *payload* dienkripsi menggunakan fungsi *base64encode*. *Base64encode* adalah metode untuk melakukan penyandian *(encoding)* terhadap data *binary* menjadi format 6-bit [14]. Hasil enkripsi *header* dan *payload* akan digabungkan dan disandi menggunakan Algoritme SHA-512 bersama dengan *secret key* yang telah ditentukan. Hasil penyandian ini akan membentuk komponen terakhir dari JWT yaitu *signature*. JWT token merupakan hasil penggabungan dari *header*, *payload* dan *signature* yang digabungkan oleh tanda titik. Token yang dihasilkan dari fungsi pembentukan tersebut seperti yang ditunjukkan pada gambar 5.

Gambar 5. Hasil JWT

Dari hasil token diatas dapat diamati bahwa token tersebut memiliki 3 bagian yang dipisahkan oleh titik. Bagian pertama yang dipisahkan oleh titip yaitu hasil enkripsi dari *header* JWT token yang berisi Algoritme hash dan tipe token. Kemudian bagian kedua yang dipisahkan oleh titik merupakan hasil enkripsi dari *payload* yang berisikan data pengguna dan informasi lainnya. Dan bagian yang terakhir merupakan *signature*.

3.2.2 Validasi JWT

Proses autentikasi jwt tidak berhenti setelah token jwt terbentuk, token jwt yang sudah terbentuk tersebut kemudian disimpan kedalam *cookies*, *session storage*, *local storage*. Ketiganya memiliki fungsi yang serupa, namun terdapat perbedaan di antara keduanya. *Local storage* memiliki kapasitas penyimpanan yang lebih besar dari 5MB tanpa memberatkan kinerja *browser*, sehingga memungkinkan pengembang web untuk menyimpan data yang lebih besar dan membuat pengalaman web yang lebih responsif di sisi klien. Sementara itu, *session storage* memiliki karakteristik di mana data yang disimpan akan terhapus ketika browser ditutup. Di sisi lain, *local storage* tidak memiliki batasan waktu kadaluarsa dan data yang disimpan di dalamnya akan tetap ada meskipun browser ditutup [15].



Gambar 6. Proses validasi JWT

Token yang dimasukan ke dalam fungsi akan dipisah lagi menjadi 3 bagian utama dengan tanda pemisah titik. Bagian yang pertama merupakan bagian dari *header*, bagian yang kedua merupakan *payload*, dan bagian yang terakhir merupakan *signature*. Dalam proses memvalidasi token yang dimasukan kedalam fungsi validasi harus menghasilkan *signature* yang

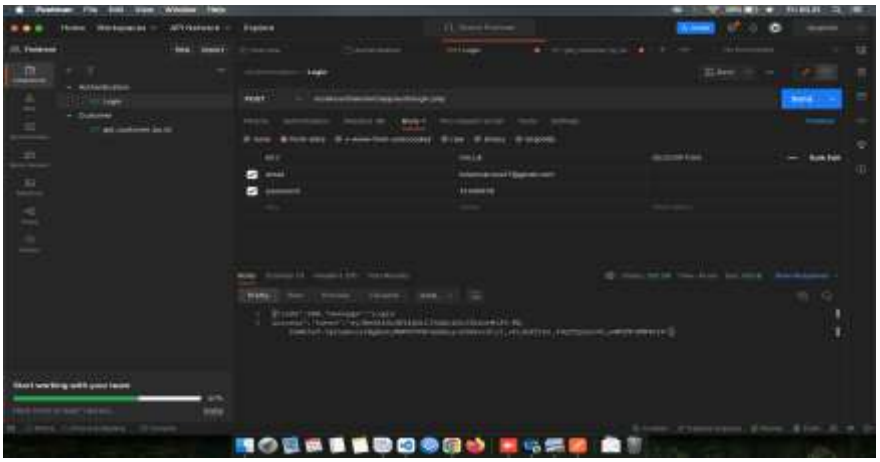
sama. Apabila hasil penyandian SHA-512 *header*, *payload* dan *secret key* tidak menghasilkan *signature* yang sama maka bisa dikatakan token tersebut tidak sesuai *(invalid)*.

JWT token akan bernilai valid apabila pembentukan ulang *signature* dari hasil penyandian *header* dan *payload* serta *secret key* sama dengan *signature* token yang dimasukan. Proses pembentukan kembali *signature* ini sangat ditentukan oleh *variabel secret key* yang telah ditentukan diawal.

3.3 Pengujian JWT

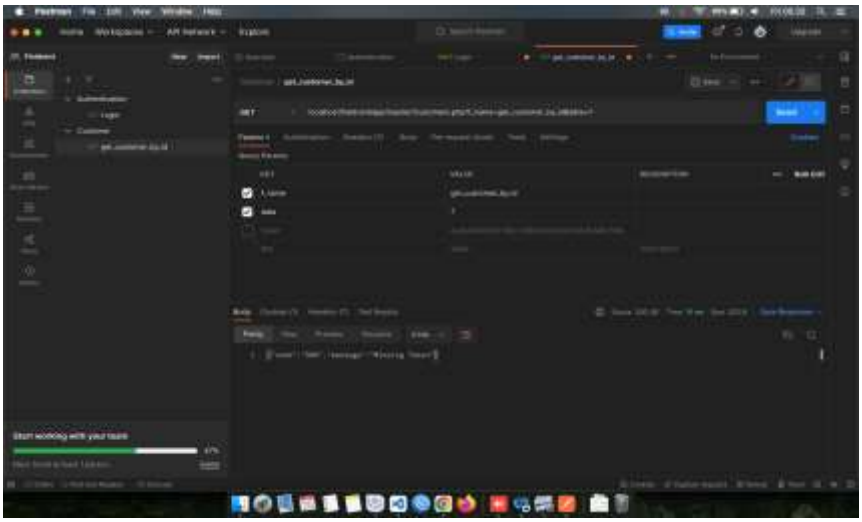
Pada Tahap pengujian ini untuk mengetahui bagaimana *response* yang akan dihasilkan aplikasi apabila melakukan *request API* tanpa menggunakan token jwt, bagaimana *response* aplikasi apabila menggunakan jwt token yang invalid dan bagaimana hasil dari memasukan token jwt yang valid. Token jwt yang valid berasal dari proses login yang benar dengan memasukan email dan *password* yang sesuai. Dalam pengujian ini peneliti menggunakan tool postman untuk mengakses API aplikasi.

Pengujian pertama adalah pengujian dari saat aplikasi tersebut melakukan autentikasi pengguna dan menghasilkan jwt token untuk disimpan kedalam *cookie browser* atau *local storage*.



Gambar 7. Pengujian API login

Pada gambar 7 ditunjukkan bahwa aplikasi akan menghasilkan *output* berbentuk json yang berisikan *code*, *message* dan token. Pada pengujian tersebut *code* 200 diartikan bahwa *request* berhasil dilakukan sehingga pengguna mendapatkan akses JWT token yang dapat dijadikan sebagai bukti kredensial pengguna.

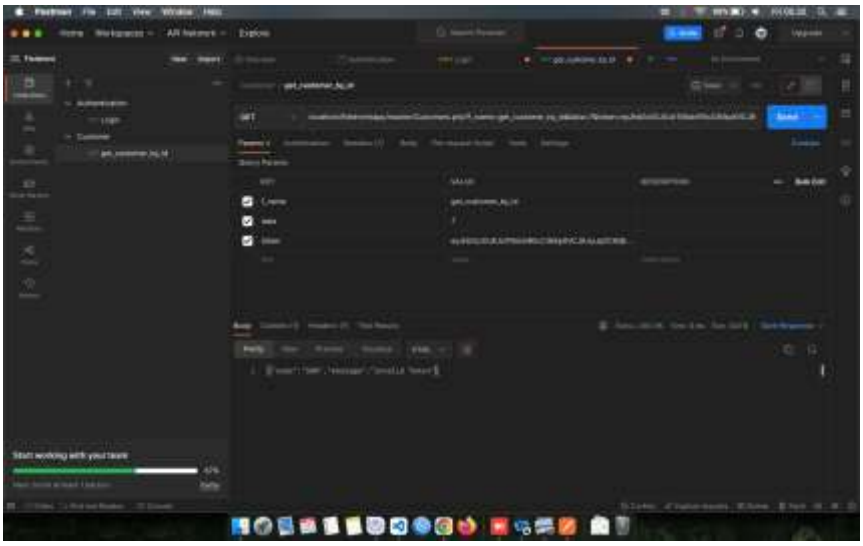


Gambar 8. Pengujan API tanpa token

Tahap selanjutnya adalah pengujian akses data *customer* tanpa menyertakan jwt token. Dalam tahap ini diharapkan pihak lain tidak dapat mengakses data *customer* tanpa adanya token sebagai bukti autentikasi pengguna.

Pada tahap pengujian ini apabila ada pengguna yang tidak menyertakan token sebagai bukti kredensial maka aplikasi akan memberikan *feedback* berupa *code 500* dan *message* : *missing token* yang mengartikan bahwa sistem gagal memproses *request* pengguna karena tidak adanya token yang disertakan.

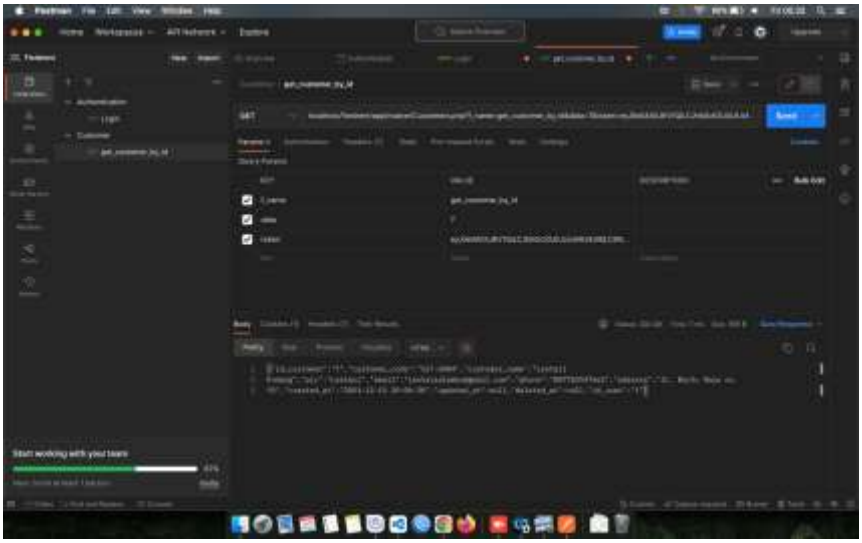
Tahap ketiga adalah menguji proses validasi token apabila token yang kita sertakan dalam parameter API merupakan hasil rekayasa pengguna. Sebagai contoh peneliti menggunakan jwt token dari aplikasi lain kemudian dimasukan kedalam parameter API. Diharapkan sistem dapat mendeteksi keabsahan dari token yang disertakan ke dalam API.



Gambar 9. Pengujian akses API dengan invalid token

Dalam pengujian ini sistem berhasil mendeteksi bahwa token yang disertakan dalam *request* bukan hasil yang di *generate* oleh aplikasi. Hal ini ditunjukkan dengan sistem memberikan *feedback* berupa *code : 500* dan *message : invalid token* yang menandakan bahwa token yang disertakan ke dalam *request* tidak valid.

Pengujian terakhir merupakan pengujian menggunakan token yang valid sehingga diharapkan API dapat memberikan umpan balik berupa data customer yang diinginkan.



Gambar 10. Pengujian akses API dengan token valid

Pada pengujian ini sistem dapat mendeteksi bahwa token yang disertakan pada API adalah token yang valid sehingga API akan memberikan umpan balik berupa data pelanggan dalam bentuk json.

4. Hasil dan Pembahasan

4.1 Komparasi dengan autentikasi konvensional

Hasil penelitian yang telah dilakukan penulis pada bab sebelumnya menggunakan aplikasi postman menunjukan bahwa hasil sistem autentikasi menggunakan jwt token sebagai sistem autentikasi dapat menanggulangi kebocoran data (*sensitive data exposure*) dibandingkan dengan autentikasi konvensional. JWT token dapat berfungsi sebagai bukti kredensial yang dimiliki pengguna setelah pengguna melakukan login ke dalam aplikasi. JWT token akan divalidasi setiap kali pengguna akan mengakses data sehingga dapat secara utuh dalam menangani kebocoran data pada sistem.

Berikut dipaparkan perbandingan antara autentikasi konvensional dan dengan JWT token yang penulis lakukan dengan sampel *API customer* pada tabel 2.

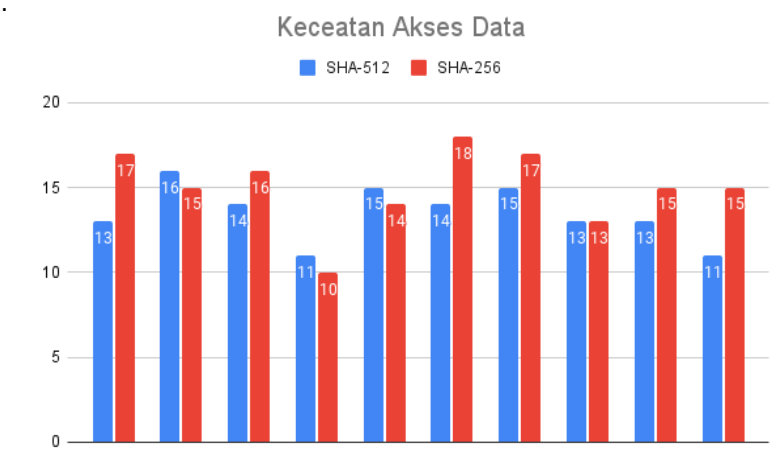
Tabel 2. Tabel perbandingan jwt token dengan konvensional

Halaman	Konvensional	JWT
Login	Hanya membatasi pengguna untuk mengakses halaman saja	Menghasilkan token jwt yang akan divalidasi setiap mengakses API
Get Data	Pengguna masih bisa mengakses API walau tidak sedang login	Pengguna tidak dapat mengakses API customer tanpa token
Insert Data	Pengguna dapat menambahkan data customer melalui API tanpa melewati proses autentikasi	Pengguna tidak dapat melakukan insert data melalui API jika tidak menyertakan token valid
Update Data	Pengguna dapat mengubah data customer melalui API tanpa melewati proses autentikasi	Pengguna tidak dapat melakukan update data jika tidak menyertakan token yang valid
Delete Data	Pengguna dapat menghapus data melalui API tanpa ada validasi token	Pengguna tidak dapat menghapus data jika tidak menyertakan token yang valid

Dalam tabel tersebut membuktikan bahwa penerapan JWT token pada sistem autentikasi akan sangat berguna dalam mengamankan data karena JWT token akan divalidasi setiap kali mengakses API.

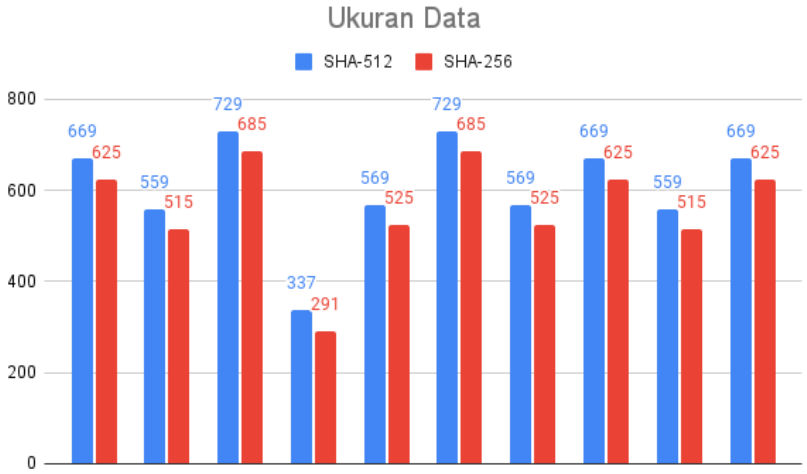
4.2 Komparasi SHA-512 dengan SHA-256

Dalam pengujian yang telah penulis lakukan guna untuk membandingkan kecepatan akses data dan besarnya ukuran data yang dihasilkan antara SHA-512 dan SHA-256 yang merupakan fungsi *hash* satu arah 256 bit [16]. Pengujian ini penulis menggunakan aplikasi postman dan telah dilakukan beberapa kali uji coba di beberapa API pada aplikasi Fieldrent. Hasil dari pengujian tersebut seperti yang ditunjukkan pada gambar 11 dan gambar 12.



Gambar 11. Grafik kecepatan akses data

Pada gambar 11 menggambarkan bahwa penggunaan Algoritme SHA-512 menghasilkan rata-rata kecepatan akses data yang lebih cepat yaitu 13,5 ms dibanding dengan rata-rata kecepatan akses data SHA-256 yang berada diangka 15 ms.



Gambar 12. Grafik ukuran data

SHA-512 menghasilkan ukuran file yang lebih besar dibanding dengan hasil yang diberikan oleh SHA-256. Dalam uji coba tersebut nilai yang dihasilkan oleh SHA-512 memiliki rata-rata ukuran 605 Byte dibanding dengan SHA-256 yang hanya menghasilkan rata-rata ukuran 561,6 Byte yang relatif lebih kecil dibanding SHA-512. Hal tersebut terjadi karena Algoritme SHA-512 memiliki panjang data yang lebih besar dibanding SHA-256.

Dari hasil pengujian yang telah disajikan diatas dapat diketahui bahwa ukuran hash SHA-512 menghasilkan ukuran file yang lebih besar dibanding dengan hasil yang diberikan oleh SHA-256. Hal tersebut terjadi karena Algoritme SHA-512 memiliki panjang data yang lebih panjang. Dengan ukurannya yang besar menghasilkan kompleksitas penyandian data menjadi lebih rumit hal ini menjadikannya lebih tahan terhadap serangan *brute-force* dan serangan kriptanalisis lainnya. SHA-512 walaupun menghasilkan data yang lebih besar namun dalam uji coba yang dilakukan penulis untuk hal kecepatan akses data SHA-512 memiliki rata-rata kecepatan akses data lebih cepat dibanding SHA-256. Sehingga penerapan sistem autentikasi JWT token dengan Algoritme SHA-512 cukup menunjang performa dan keamanan aplikasi Fieldrent. Hal ini mendukung penelitian sebelumnya yang mengatakan bahwa penerapan sistem autentikasi JWT token dapat meningkatkan keamanan aplikasi.

5. Simpulan

Penerapan JWT token sebagai sistem autentikasi keamanan aplikasi Fieldrent terbukti lebih aman dibandingkan dengan sistem autentikasi konvensional seperti yang disajikan pada tabel 2. Penggunaan Algoritme SHA-512 mendukung keamanan JWT token dalam penyandian data sensitif sebagaimana hasil uji coba penelitian diatas, penerapan SHA-512 sebagai Algoritme JWT token memiliki kompleksitas penyandian data yang rumit sehingga menjadikannya lebih tahan terhadap serangan *brute-force* dan memiliki kecepatan akses data yang relatif lebih cepat dibanding Algoritme SHA-256.

Daftar Referensi

[1] F. Kwarto and M. Angsito, "Pengaruh Cyber Crime Terhadap Cyber Security Compliance Di Sektor Keuangan," *J. Akunt. Bisnis*, vol. 11, no. 2, pp. 99–110, 2018, doi: 10.30813/jab.v11i2.1382.

[2] M. I. D. Putra, "Keamanan Siber di Indonesia: Apakah Kita Benar-Benar Aman di Dunia Siber?," *cfds.fisipol.ugm.ac.id*, 2023.

[3] Edward Chandra, "SENSITIVE DATA EXPOSURE," *mti.binus.ac.id*, 2018.

[4] M. Ipdal, "Analisa Metode SHA-512 Untuk Tanda Tangan Digital Pada File Video," *Journal of Informatics Management and Information Tech.,* vol. 1 no. 1, pp. 23–29, 2021.

[5] A. Umarjati and A. Wibowo, "Implementasi JWT pada Aplikasi Presensi dengan Validasi Fingerprint ," *J. RESTI (Rekayasa Sist. dan Teknol. Informasi)*, vol. 1, no. 10, pp. 1085–1091, 2021.

[6] B. Di and P. Multisindo, "IMPLEMENTASI RESTFUL DENGAN JWT UNTUK BOOKING BARANG DI PRIMAJAYA MULTISINDO," *Senafti*, vol. 1, no. 1, pp. 1040–1047, 2022.

[7] Z. Abidin, A. Wijaya, D.Pasha, "Aplikasi Stemming Kata Bahasa Lampung Dialek Api Menggunakan Pendekatan Brute-Force dan Pemrograman C#," *Jurnal Media Informatika Budidarma,* vol. 5, no. 1, pp. 1–8, 2021.

[8] F. C. Ramdani, A. rahmatulloh, R. N. Shofa, "Implementasi JSON Web Token pada Authentication dengan Algoritme HMAC SHA-256 ," *SISTEMASI (Jurnal Sistem Informasi),* vol. 12 no. 1, pp. 1s94-205, 2022.

[9] A. W. P.. Putra, A. Bhawiyuga, M. Data, "Implementasi Autentikasi JSON Web Token (JWT) Sebagai Mekanisme Autentikasi Protokol MQTT Pada Perangkat NodeMCU," *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer,* vol. 2, no. 2, pp 584-593, 2018.

[10] A. Setiawan, A. i. Purnamasari "Implementasi JSON Web Token Berbasis Algoritme SHA-512 untuk Otentikasi Aplikasi BatikKita," J. Resti, vol 4, no. 6, pp 1036-1045, 2020.

[11] I. Priyadi and R. J. Wellem Sagay, "Perancangan dan Implementasi Aplikasi Pasar Komoditas Andalan Desa (PAKADES) Berbasis Mobile Android untuk Petani Desa," *J. Sist. Inf.*, vol. 1, no. 2, pp. 9–14, 2019.

[12] N. Rasyada, "SHA-512 Algorithm on Json Web Token for Restful Web Service-Based Authentication," *J. Appl. Data Sci.*, vol. 3, no. 1, pp. 33–43, 2022, doi: 10.47738/jads.v3i1.51.

[13] B. Satria, A. Kusyanti, and W. Yahya, "Implementasi Algoritme Blake2s pada JSON Web Token (JWT) sebagai Algoritme Hashing untuk Mekanisme Autentikasi Layanan REST-API," *J. Pengemb. Teknol. Inf. dan Ilmu Komput. Univ. Brawijaya*, vol. 2, no. 12, pp. 6269–6276, 2018.

[14] M. F. Arif, and M. Misdram, "Implementasi Enkripsi Url Pada Website Menggunakan Metode Base64 Dan Rotation13," *J. Spirit*, vol. 12, no. 1, pp. 20–25, 2020.

[15] J. Triyono, "Implementasi Localstorage pada Pemrograman Client Berbasis JSON," Simposium Nasional RAPI, vol. 18, no. 13, pp. 371–379, 2019.

[16] S. Sulastri and R. D. M. Putri, "Implementasi Enkripsi Data Secure Hash Algorithm (SHA-256) dan Message Digest Algorithm (MD5) pada Proses Pengamanan Kata Sandi Sistem Penjadwalan Karyawan," *J. Tek. Elektro*, vol. 10, no. 2, pp. 70–74, 2018.