ISSN: 2598-8565 (media cetak)
ISSN: 2620-4339 (media online)

IMPLEMENTASI API RESTFUL DENGAN JSON WEB TOKEN (JWT) PADA APLIKASI E-COMMERCE THRIFTY SHOP UNTUK OTENTIKASI DAN OTORISASI PENGGUNA

Ahmad Yahya Nashikhuddin✉, Jamilah Karaman, Yovi Litanianda
Teknik Informatika, Universitas Muhammadiyah Ponorogo, Indonesia
Email: ahmad.yahya.n@gmail.com

DOI: https://doi.org/10.46880/jmika.Vol7No2.pp239-246

ABSTRACT

User authentication and authorization play a vital role in securing sensitive data within applications. E-commerce applications, in particular, require robust authentication and authorization methods to safeguard the confidentiality and integrity of exchanged data. JSON Web Token (JWT) has emerged as a popular authentication mechanism for securing data transmission over networks. This paper explores the implementation of JWT in a RESTful API to achieve user authentication and authorization in an E-commerce application. The objective of this study is to analyze the implementation of JWT in a RESTful API for user authentication and authorization within an E-commerce context. The detailed implementation of JWT in the Thrifty Shop application is discussed, encompassing its utilization for user authentication and authorization. Additionally, the benefits derived from employing JWT in the Thrifty Shop E-commerce application are explored. In conclusion, the implementation of JSON Web Token (JWT) in a RESTful API proves to be an effective approach for user authentication and authorization in E-commerce applications. The use of JWT ensures that only authenticated users gain access to sensitive data, thus enhancing the security of the Thrifty Shop E-commerce application. This implementation can serve as a model for other E-commerce applications seeking to bolster their security measures.
Keyword: JSON Web Token (JWT), E-Commerce Application, Authentication, Authorization, RESTful API.

ABSTRAK

Otentikasi dan otorisasi pengguna merupakan aspek penting dalam setiap aplikasi yang mengelola data pengguna yang sensitif. Aplikasi e-commerce, khususnya, memerlukan metode otentikasi dan otorisasi pengguna yang aman dan dapat diandalkan untuk menjaga kerahasiaan dan integritas data yang ditransfer. JSON Web Token (JWT) adalah metode otentikasi yang umum digunakan untuk mengamankan data yang dikirim melalui jaringan. Dalam penelitian ini, kami menjelajahi implementasi JWT dalam API RESTful untuk otentikasi dan otorisasi pengguna pada aplikasi e-commerce. Tujuan dari penelitian ini adalah untuk menganalisis implementasi JWT dalam API RESTful untuk otentikasi dan otorisasi pengguna pada aplikasi e-commerce. Implementasi JWT dalam aplikasi Thrifty Shop akan dibahas secara detail, termasuk penggunaannya untuk otentikasi dan otorisasi pengguna. Selain itu, manfaat penggunaan JWT dalam aplikasi e-commerce Thrifty Shop juga akan dieksplorasi. Sebagai kesimpulan, implementasi JSON Web Token (JWT) dalam API RESTful adalah metode yang efektif untuk otentikasi dan otorisasi pengguna dalam aplikasi e-commerce. Penggunaan JWT membantu memastikan bahwa hanya pengguna yang terotentikasi yang mendapatkan akses ke data sensitif, sehingga meningkatkan keamanan aplikasi e-commerce Thrifty Shop. Implementasi JWT ini dapat menjadi contoh bagi aplikasi e-commerce lainnya yang ingin meningkatkan langkah-langkah keamanan mereka.
Kata Kunci: JSON Web Token (JWT), Aplikasi E-Commerce, Otentikasi, Otorisasi, RESTful API.

PENDAHULUAN

Otentikasi dan otorisasi adalah aspek penting dalam pengembangan aplikasi, terutama bagi aplikasi *e-commerce* yang membutuhkan tingkat keamanan yang tinggi dalam mengelola data pengguna. *JWT* adalah token berbentuk *string* panjang *random* yang gunanya untuk melakukan autentikasi sistem dan pertukaran informasi (Putra, Bhawiyuga, & Data, 2018). Sedangkan *REST* merupakan arsitektur *web service* yang bersifat *client server* dimana *client* melakukan *request* kepada *server* kemudian *server* memproses *request* dan mengembalikan *response*. *RESTful web service* meurpakan sebutan untuk aplikasi *web* yang menggunakan arsitektur dari *REST*. Dalam menggunakan teknologi *REST* dibantu dengan sebuah tools yang bernama *API* (*Application Programming*

Interface) berbasis *web*site. *API* secara umum terdiri dari 2 bagian, yaitu *server* sebagai penyedia data dan client yang dapat melakukan *request* data. *REST API* adalah *API* berbasis *web*site yang menggunakan teknologi *REST* dan menggunakan format *JSON (JavaScript Object Notation),* yaitu sebuah format pertukaran data yang bisa digunakan baik pada *front-end* maupun *back-end* dari aplikasi *web*site maupun sebuah *service* (Wardhana, Arwani, & Rahayudi, 2020). Jadi intinya, *API RESTful* menyediakan antarmuka komunikasi yang terbuka antara aplikasi *web* dan sumber daya yang dikelola, memberikan fleksibilitas dalam pengembangan aplikasi.

Pada penelitian ini, peneliti melakukan implementasi *API RESTful* dan *JSON Web* Token (*JWT*) pada aplikasi *e-commerce Thrifty Shop* untuk otentikasi dan otorisasi pengguna. Tujuan dari penelitian ini adalah untuk menganalisis penggunaan *API RESTful* dan *JSON Web* Token (*JWT*) pada aplikasi *e-commerce Thrifty Shop* untuk meningkatkan keamanan dan performa aplikasi.

Penggunaan *API RESTful* dan *JSON Web* Token (*JWT*) pada aplikasi *e-commerce Thrifty Shop* telah memberikan beberapa manfaat, seperti peningkatan keamanan dan performa aplikasi. Selain itu, implementasi *API RESTful* dan *JSON Web* Token (*JWT*) pada aplikasi *e-commerce Thrifty Shop* juga dapat menjadi model bagi aplikasi *e-commerce* lainnya dalam meningkatkan keamanan dan performa aplikasi mereka.

Dalam kesimpulannya, implementasi *API RESTful* dengan *JSON Web* Token (*JWT*) pada aplikasi *e-commerce Thrifty Shop* adalah metode yang efektif untuk meningkatkan keamanan dan performa aplikasi. Penggunaan *API RESTful* memungkinkan pengembang aplikasi untuk memanfaatkan teknologi terbaru dan memper*oleh* fleksibilitas dalam pengembangan aplikasi. Sementara itu, penggunaan *JSON Web* Token (*JWT*) memastikan bahwa hanya pengguna yang terotentikasi yang diberikan akses ke data sensitif, meningkatkan keamanan aplikasi *e-commerce Thrifty Shop*.

TINJAUAN PUSTAKA

Representational State Transfer (*REST*)

REST merupakan arsitektur *web service* yang dikembangkan dari beberapa gaya arsitektur berbasis jaringan yang sering diterapkan dalam layanan berbasis *web* (Gunawan & Rahmatulloh, 2019).

Pada arsitektur *REST*, sumber daya (resource) merupakan elemen utama yang diidentifikasi melalui *URI (Uniform Resource Identifier).* Sumber daya ini direpresentasikan dalam format tertentu seperti *JSON* atau *XML* (Edy, Ferdiansyah, Pramusinto, & Waluyo, 2019). dapat diakses dan dimanipulasi oleh klien melalui serangkaian operasi standar *HTTP* seperti *GET* untuk membaca data dari database, *POST* untuk membaca data dari database, *PUT* untuk memperbarui data, dan *DELETE* digunakan untuk menghapus data dari database (Safitri & Putro, 2021).

Keuntungan utama dari penggunaan arsitektur *REST* adalah kemampuan untuk memisahkan antara tampilan dan data, sehingga memungkinkan pengembangan aplikasi yang lebih fleksibel dan terdistribusi. Selain itu, *REST* juga memudahkan integrasi antara sistem dan aplikasi yang berbeda, karena memungkinkan sistem tersebut untuk berkomunikasi melalui antarmuka yang terbuka dan standar. Berikut adalah secara umum metode yang digunakan pada *REST* ditampilkan pada Tabel 1.

Tabel 1. Metode *REST*

Resource	***Method***			
	Get	***Post***	***Put***	***Delete***
/product	Mendapatkan list semua produk	Membuat list produk baru	Memperbarui data suatu produk	Menghapus semua produk
/product/1	Mendapatkan data suatu produk berdasarkan ID produk	Membuat produk baru dan memasukkan ke dalam list	Memperbarui data produk jika ID produk ada, jika tidak akan *error*	Menghapus data produk

JSON Web Token (JWT)

JWT merupakan sebuah token berbentuk string yang terdiri dari tiga bagian yaitu : *header*, *payload* dan *signature* yang digunakan untuk proses otentikasi dan pertukaran informasi (Rispian, Rahmatullah, & Sari, 2019). Token ini biasanya digunakan untuk autentikasi dan otorisasi pengguna pada aplikasi *web* dan *mobile*.

Berikut adalah karakteristik dari *JSON Web* Token (*JWT*):

1. *Header*
   Header biasanya terdiri dari dua bagian: jenis token, *JWT*, dan algoritma *hashing* yang digunakan, seperti *HMAC, RSA*, atau algoritma lainnya (Putra et al., 2018).

```
{
        "typ": "JWT",
        "alg": "HMAC512"

}
```

2. *Payload*
   Bagian *payload JWT* berisi klaim-klaim atau informasi yang ingin disampaikan dalam token. Klaim-klaim ini dapat berisi informasi pengguna, hak akses, waktu kadaluwarsa, atau klaim kustom lainnya. Terdapat tiga jenis klaim standar dalam *JWT*: klaim terdaftar (*registered claims*), klaim pribadi (*private claims*), dan klaim publik (*public claims)*. Klaim terdaftar termasuk "*iss*" (*issuer*), "*exp*" (*expiration time*), "*sub*" (*subject*), dan lain-lain (Jones, Bradley, & Sakimura, 2015). Klaim pribadi adalah klaim kustom yang ditentukan oleh pengguna. Klaim publik adalah klaim yang telah ditentukan dan dapat digunakan oleh siapa pun.

3. *Signature*
   Bagian *signature JWT* digunakan untuk memverifikasi keaslian token. *Signature* hasil dari Hash atau gabungan dari isi encode Header dan Payloadnya lalu ditambahkan kode secretnya. Signature ini berguna untuk memverifikasi bahwa header maupun payload yang ada dalam token tidak berubah dari nilai aslinya (karena untuk membuat payload dan header palsu itu cukup mudah). Signature-nya sendiri tidak mungkin dapat diakali, karena sudah dalam berbentuk hash yang mana adalah fungsi satu arah (tidak dapat dikembalikan ke nilai semula), dan meski kita tahu algoritma hashing-nya, kita juga memerlukan secret key yang mana hanya si pembuat aplikasi yang tahu (Hadyan, 2021).

```
HMACSHA256(base64UrlEnkode(header) + '.' + base64UrlEnkode(payload), secretKey)
```

Dengan menggabungkan ketiga bagian tersebut, kita dapat membentuk *JWT* yang aman dan dapat diverifikasi. *Header* dan *payload* dienkode menggunakan *Base64* URL enkoding sebelum digabungkan dengan tanda titik (.) untuk membentuk token *JWT* yang lengkap atau *verify signature* (Sitorus, Kusyanti, & Bhawiyuga, 2020).

METODE PENELITIAN

Gambar 1. *Metodologi Penelitian*

Gambar 1 menunjukan alur metodologi penelitian. Metode penelitian ini bertujuan untuk merancang dan mengimplementasikan penggunaan *JWT* sebagai mekanisme otentikasi dan otorisasi pengguna dalam aplikasi *e-commerce Thrifty Shop*.

Metode studi literatur digunakan untuk memper*oleh* pemahaman yang mendalam tentang konsep dasar *JSON Web* Token (*JWT*), prinsip autentikasi, otorisasi, dan keamanan pada aplikasi *e-commerce*. Dalam tahap ini, dilakukan pencarian dan analisis terhadap literatur ilmiah, artikel jurnal, buku, dan sumber informasi terpercaya lainnya yang relevan dengan penggunaan *JWT* pada aplikasi *e-commerce*. Studi literatur ini menjadi dasar peneliti untuk memahami konsep-konsep yang terkait dengan implementasi *JWT* pada *Thrifty Shop*. Studi literatur yang dilakukan meliputi :

1. Rohmat Gunawan., & Alam Rahmatulloh. (2019). *JSON Web* Token (*JWT*) untuk Authentication pada Interoperabilitas Arsitektur berbasis *REST*ful *Web Service* (Gunawan & Rahmatulloh, 2019).
2. Andri Warda Pratama *Put*ra., Adithya Bhawiyuga., & Mahendra Data. (2018). Implementasi

Autentikasi *JSON Web* Token (*JWT*) Sebagai Mekanisme Autentikasi Protokol MQTT Pada Perangkat NodeMCU (Putra et al., 2018).

3. Bagus Satria., Ari Kusyanti., & Widhi Yahya. (2018). Implementasi Algoritme Blake2s pada *JSON Web* Token (*JWT*) sebagai Algoritme Hashing untuk Mekanisme Autentikasi Layanan *REST-API* (Wiguna, Kusyanti, & Yahya, 2018).

Kemudian analisis kebutuhan meliputi kebutuhan keamanan data, mekanisme otentikasi yang kuat, mekanisme otorisasi yang tepat, skalabilitas sistem, dan integrasi dengan komponen lain. Analisis ini membantu dalam merancang solusi yang sesuai dengan kebutuhan aplikasi *e-commerce Thrifty Shop* untuk memastikan otentikasi dan otorisasi yang aman dan efektif bagi pengguna.

Pada tahap perancangan sistem, peneliti akan merancang sistem otentikasi dan otorisasi menggunakan *JWT* pada aplikasi *e-commerce Thrifty Shop*. Perancangan ini mencakup pemodelan struktur token *JWT*, pemilihan algoritma enkripsi yang sesuai, integrasi dengan sistem *backend*, serta perancangan mekanisme validasi token. Dalam perancangan ini, peneliti akan mempertimbangkan keamanan, skalabilitas, dan kinerja sistem.

Implementasi merupakan tahap di mana peneliti menerapkan rancangan sistem otentikasi dan otorisasi menggunakan *JWT* pada aplikasi *e-commerce Thrifty Shop*. Peneliti akan mengimplementasikan logika penggunaan *JWT*, pengelolaan token, proses validasi, serta fitur-fitur terkait keamanan lainnya pada sistem *backend* aplikasi.

HASIL DAN PEMBAHASAN

Analisis Kebutuhan

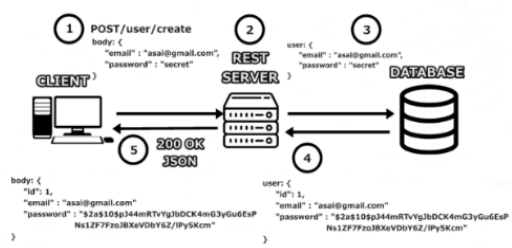
Dalam tahap analisis kebutuhan, peneliti melakukan identifikasi dan analisis terhadap kebutuhan otentikasi dan otorisasi pengguna dalam aplikasi *e-commerce Thrifty Shop*. Tujuan dari analisis kebutuhan ini adalah untuk memahami fitur-fitur yang perlu diimplementasikan dalam penggunaan *JSON Web* Token (*JWT*) sebagai mekanisme otentikasi dan otorisasi pengguna.

Peneliti melakukan studi literatur dan juga berkomunikasi dengan tim pengembang aplikasi *Thrifty Shop* untuk mengumpulkan informasi tentang kebutuhan pengguna yang relevan. Beberapa aspek yang dianalisis dalam analisis kebutuhan antara lain:

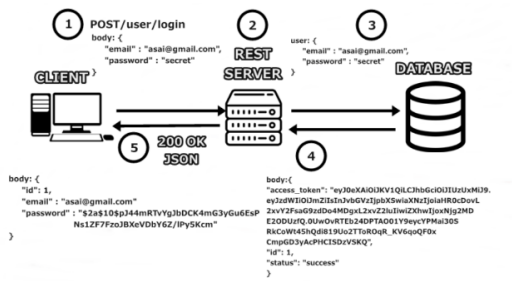
1. Sistem *Login*: Peneliti menganalisis kebutuhan terkait proses *login* pengguna ke dalam aplikasi *Thrifty Shop*. Ini meliputi validasi identitas pengguna, autentikasi, serta pemberian akses ke fitur-fitur yang sesuai dengan peran dan izin pengguna.
2. Pengaturan Hak Akses: Peneliti mengidentifikasi kebutuhan terkait pengaturan hak akses pengguna. Hal ini mencakup pengaturan peran (*role*) pengguna dan penentuan aksesibilitas terhadap fitur-fitur tertentu berdasarkan peran yang dimiliki.
3. Validasi Token: Peneliti menganalisis kebutuhan terkait validasi token *JWT* yang digunakan untuk mengautentikasi pengguna. Ini meliputi verifikasi keaslian token, pengecekan masa berlaku token, serta keabsahan tanda tangan digital pada token.
4. Pengelolaan Sesi: Peneliti mempertimbangkan kebutuhan pengelolaan sesi pengguna yang menggunakan *JWT*. Ini mencakup kebutuhan terkait pembuatan, pemutakhiran, dan penghapusan sesi pengguna.

Perancangan Sistem

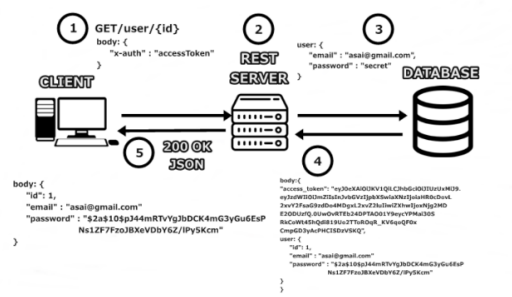
Dalam penelitian yang dilakukan perancangan system autentikasi dibagi menjadi 4 skema yaitu *register*, *login*, *user information* dan *logout*



Gambar 2 Skema *Register*

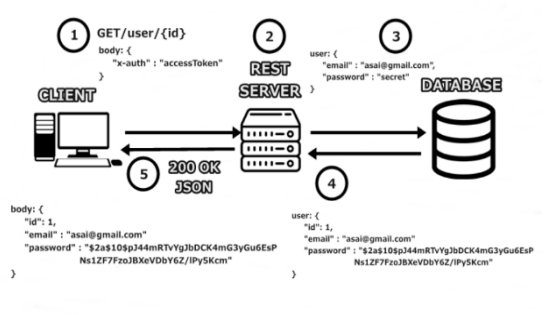


Gambar 3 Skema *Login*



Gambar 4 Skema *User Information*

ISSN: 2598-8565 (media cetak)
ISSN: 2620-4339 (media online)



Gambar 5 Skema *Logout*

Pengujian

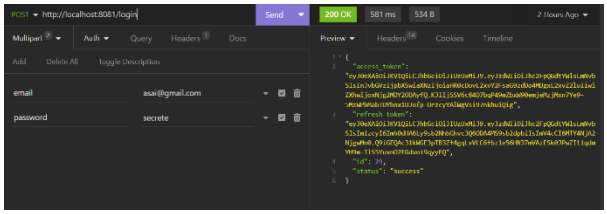
Pengujian dilakukan untuk memastikan keberhasilan implementasi *JWT* dalam aplikasi. Beberapa jenis pengujian yang dapat dilakukan meliputi:

1. Pengujian Unit: Melibatkan pengujian terhadap setiap komponen atau fungsi yang terlibat dalam penggunaan *JWT*, seperti pembuatan token, verifikasi token, dan manipulasi token. Pengujian ini bertujuan untuk memastikan bahwa setiap komponen berjalan dengan baik secara individual.
2. Pengujian Fungsional: Melibatkan pengujian fungsionalitas keseluruhan aplikasi dengan menggunakan *JWT* untuk otentikasi dan otorisasi pengguna. Pengujian ini mencakup simulasi skenario penggunaan yang berbeda, termasuk pengujian otentikasi sukses dan gagal, otorisasi akses yang sesuai, serta pengelolaan token yang valid dan kedaluwarsa.
3. Pengujian Waktu Autentikasi: jenis pengujian yang dilakukan untuk mengukur waktu yang dibutuhkan oleh sistem dalam proses otentikasi pengguna. Pengujian ini bertujuan untuk mengevaluasi kinerja sistem dalam memproses permintaan otentikasi dan memberikan respons kepada pengguna.

Pengujian Unit

Pengujian unit dibagi menjadi skenario yaitu pengambilan token baru, validasi token, Pengujian Otorisasi, *Handling Error.*

1. Pengambilan Token Baru
   Memastikan bahwa fungsi pembuatan token *JWT* berfungsi dengan benar. Pengujian ini melibatkan mengirimkan data pengguna yang valid ke fungsi pembuatan token, kemudian memeriksa apakah token yang dihasilkan sesuai dengan format dan informasi yang diharapkan.

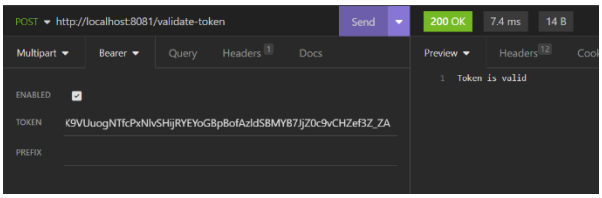


Gambar 6. Autentikasi *JWT*

Menggunakan email " asai@gmail.com" dan *password* " *secrete*" dihasilkan *JWT* berupa.

Gambar 7. *JWT* Token

2. Validasi Token
   Memastikan bahwa fungsi validasi token berhasil mengenali token *JWT* yang valid dan menolak token yang tidak valid. Pengujian ini melibatkan mengirimkan token yang valid dan token yang tidak valid ke fungsi validasi, kemudian memeriksa apakah hasilnya sesuai dengan harapan.

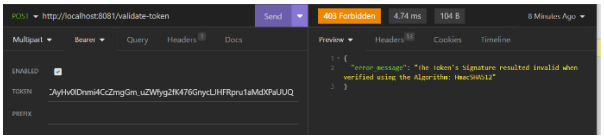


Gambar 8. Validasi Token Valid

Ketika token valid maka akan menghasilkan out*put* "Token is valid" seperti pada Gambar 8.

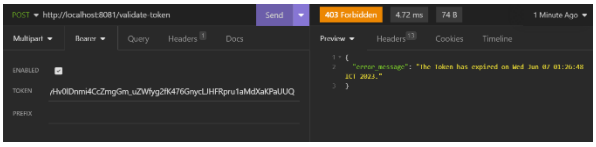
3. *Handling Error*
   Memastikan bahwa aplikasi mampu menangani situasi *error* dengan baik. Pengujian ini melibatkan mengirimkan permintaan dengan token yang tidak valid atau permintaan yang tidak lengkap, dan memeriksa apakah aplikasi memberikan respons *error* yang sesuai.

ISSN: 2598-8565 (media cetak)
ISSN: 2620-4339 (media online)



Gambar 9. Validasi Token Tidak Valid

Pada Gambar 9. token yang tidak valid akan menghasilkan out*put "The Token's Signature resulted invalid when verified using the Algorithm: HmacSHA512*" sedangkan untuk token yang kadaluarsa maka akan menhasilkan out*put "The Token has expired on Wed Jun 07 01:26:48 ICT 2023."* Seperti pada Gambar 10.



Gambar 10. Token Ketika Kadaluarsa

Pengujian Fungsionalitas Sistem

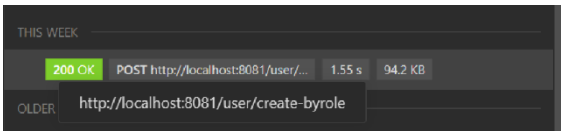
Pengujian fungsionalitas sistem dilakukan untuk memastikan bahwa fungsionalitas sistem *REST API* yang dibangun dapat berjalan tanpa menemukan kendala atau *error* (Wiguna et al., 2018).

Terdapat 6 pengujian skema yaitu

1. Pengujian Registrasi Pengguna
2. Pengujian Otentikasi Pengguna
3. Pengujian *Protected Resource Access*
4. Pengujian *Unprotected Resource Access*
5. Pengujian Akses Token *Kadaluarsa*
6. Pengujian Akses Token Tidak Valid

Pada percobaan pertama dilakukan registrasi dengan tiga kali percobaan yaitu :

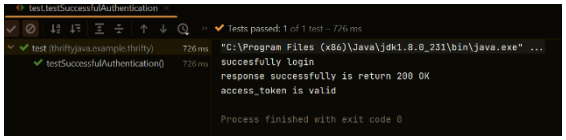
1. Mengirimkan permintaan *POST* ke endpoint *user/create-byrole* dengan data pengguna yang valid.
2. Memeriksa bahwa respons kembali dengan kode status 200 *OK* dan memberikan token akses
3. Memeriksa bahwa pengguna berhasil terdaftar dalam basis data.
4.



Gambar 11. Hasil Pengujian Registrasi

Pada percobaan ke-dua dilakukan otentikasi dengan tiga kali percobaan :

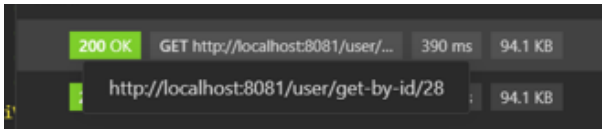
1. Mengirimkan permintaan *POST* ke endpoint */login* dengan informasi otentikasi pengguna yang valid.
2. Memeriksa bahwa respons kembali dengan kode status 200 *OK* dan memberikan token akses.
3. Memeriksa bahwa token akses valid dan dapat digunakan untuk mengakses sumber daya yang diotorisasi.



Gambar 12. Hasil Pengujian Otentikasi

Percobaan ke-tiga dilakukan pengujian *Protected Resource Access* dengan tiga kali percobaan:

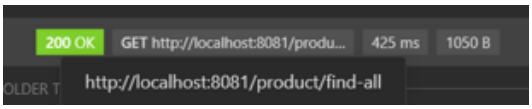
1. Mengirimkan permintaan *GET* atau *POST* ke endpoint yang memerlukan otentikasi, seperti *user/get-by-id*.
2. Memasukkan token akses yang valid dalam *header* permintaan.
3. Memeriksa bahwa respons kembali dengan kode status 200 *OK* dan memberikan data sumber daya yang diharapkan.



Gambar 13. Hasil Pengujian *Protected Resource Access*

Percobaan ke-empat dilakukan pengujian *Unprotected Resource Access* dengan dua kali percobaan :

1. Mengirimkan permintaan *GET* atau *POST* ke endpoint yang tidak memerlukan otentikasi, seperti */product/find-all*.
2. Memeriksa bahwa respons kembali dengan kode status 200 *OK* dan memberikan data sumber daya yang diharapkan.

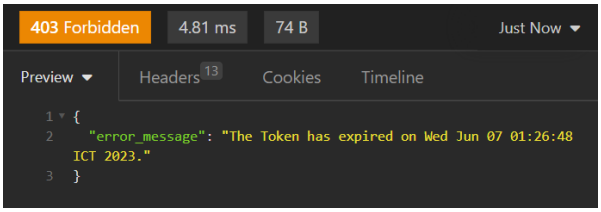


Gambar 14. Hasil Pengujian *Unprotected Resource Access*

Percobaan ke-lima dilakukan pengujian akses token kadaluarsa dengan dua percobaan :

1. Mengirimkan permintaan dengan token akses yang telah kedaluwarsa.
2. Memeriksa bahwa respons kembali dengan kode status 401 *Unauthorized* atau *403 Forbidden*.

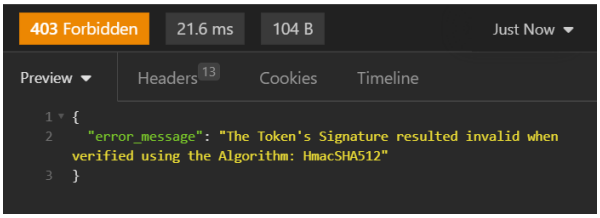
ISSN: 2598-8565 (media cetak)
ISSN: 2620-4339 (media online)



Gambar 15. Hasil Pengujian Akses Token Kadaluarsa

Percobaan ke-enam dilakukan pengujian akses token tidak valid dengan dua percobaan :

1. Mengirimkan permintaan dengan token akses yang tidak valid atau telah dimanipulasi.
2. Memeriksa bahwa respons kembali dengan kode status 401 *Unauthorized* atau 403 *Forbidden*.



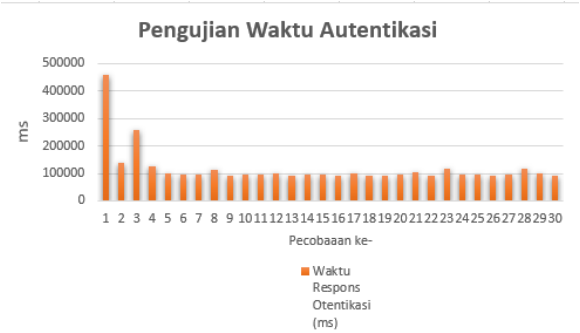
Gambar 16. Hasil Pengujian Akses Token Tidak Valid

Pengujian Waktu Autentikasi

Pengujian Waktu Autentikasi membantu mengidentifikasi keterlambatan atau kegagalan dalam proses otentikasi pengguna, sehingga dapat dilakukan perbaikan atau optimasi untuk meningkatkan kinerja sistem. Prosedur pengujian waktu autentikasi dilakukan sebanyak 30 kali percobaan. Pada akhir percobaan disimpan nilai keseluruhan percobaan dan rata rata waktu yang dibutuhkan kemudian hasilnya akan ditampilkan. Hasil percobaan akan ditampilkan pada tabel 2.

Tabel 2. Tabel Hasil Pengujian Waktu Autentikasi

Percobaan ke-	Waktu Respons Otentikasi (*ms*)	Percobaan ke-	Waktu Respons Otentikasi (*ms*)
1	458.957	16	90.991
2	137.389	17	98.996
3	257.185	18	92.380
4	124.608	19	93.494
5	102.160	20	94.815
6	96.293	21	105.181
7	95.777	22	89.896
8	113.778	23	115.421
9	92.185	24	94.885
10	96.962	25	96.952
11	94.165	26	93.070
12	99.240	27	96.267
13	93.517	28	117.104
14	95.424	29	100.105
15	97.605	30	92.438
Rata - Rata		101.792 *ms*	
Total		3,053.764 *ms*	



Gambar 17. Grafik Hasil Pengujian Waktu Autentikasi

Berdasarkan percobaan yang dilakukan pada Tabel 2 didapatkan waktu respons otentikasi yang beragam, mulai dari 89.896 *ms* hingga 117.104 *ms*. Rata-rata waktu respons otentikasi adalah 101.792 *ms*. Total waktu respons otentikasi dalam semua percobaan adalah 3,053.764 *ms*.

KESIMPULAN

Dalam penelitian ini, implementasi *API RESTful* dengan *JSON Web* Token (*JWT*) pada aplikasi *E-commerce Thrifty Shop* untuk otentikasi dan otorisasi pengguna memberikan solusi yang efektif dan aman dalam mengelola pengguna dan sumber daya aplikasi. Dengan menggunakan JWT, aplikasi ini akan memastikan bahwa pengguna yang menggunakan layanan tersebut telah diotentikasi dan memiliki izin yang tepat untuk mengakses sumber daya tertentu. JWT akan berperan sebagai mekanisme untuk mengenkripsi dan menyampaikan informasi otentikasi secara aman antara pengguna dan server.

DAFTAR PUSTAKA

Edy, E., Ferdiansyah, F., Pramusinto, W., & Waluyo, S. (2019). Pengamanan Restful API menggunakan JWT untuk Aplikasi Sales Order. *Jurnal RESTI (Rekayasa Sistem Dan Teknologi Informasi)*, *3*(2), 106–112. https://doi.org/10.29207/resti.v3i2.860

Gunawan, R., & Rahmatulloh, A. (2019). JSON Web Token (JWT) untuk Authentication pada Interoperabilitas Arsitektur berbasis RESTful Web Service. *Jurnal Edukasi Dan Penelitian Informatika (JEPIN)*, *5*(1), 74. https://doi.org/10.26418/jp.v5i1.27232

Hadyan, P. (2021). Kenalan Yuk Dengan JSON Web Token (JWT). Retrieved from https://codepolitan.com/blog/kenalan-yuk-dengan-json-web-token-jwt

Jones, M., Bradley, J., & Sakimura, N. (2015). *JSON Web Token (JWT)*. https://doi.org/10.17487/RFC7519

Putra, A. W. P., Bhawiyuga, A., & Data, M. (2018). Implementasi Autentikasi JSON Web Token (JWT) Sebagai Mekanisme Autentikasi Protokol MQTT Pada Perangkat NodeMCU. *Jurnal Pengembangan Teknologi Informasi Dan Ilmu Komputer*, *2*(2), 584–593.

Rispian, P., Rahmatullah, A., & Sari, R. H. M. I. (2019). *Implementasi Authentication dengan JSON Web Token (JWT) pada Laravel Single Page Application*.

Safitri, R. K., & Putro, H. P. (2021). Implementasi REST API untuk Komunikasi Antara ReactJS dan NodeJS (Studi Kasus : Modul Manajemen User Solusi247). *AUTOMATA*, *2*(1), 1–5.

Sitorus, N. F., Kusyanti, A., & Bhawiyuga, A. (2020). Implementasi Autentikasi Berbasis Token Menggunakan Platform-Agnostic Security Tokens (PASETO) Sebagai Mekanisme Autentikansi RESTful API. *Jurnal Pengembangan Teknologi Informasi Dan Ilmu Komputer*, *4*(11), 3947–3955.

Wardhana, W. G., Arwani, I., & Rahayudi, B. (2020). Implementasi Teknologi Restful Web Service Dalam Pengembangan Sistem Informasi Perekaman Prestasi Mahasiswa Berbasis Website (Studi Kasus: Fakultas Teknologi Pertanian Universitas Brawijaya). *Jurnal Pengembangan Teknologi Informasi Dan Ilmu Komputer*, *4*(2), 680–689.

Wiguna, B. S., Kusyanti, A., & Yahya, W. (2018). Implementasi Algoritme Blake2s pada JSON Web Token (JWT) sebagai Algoritme Hashing untuk Mekanisme Autentikasi Layanan REST-API. *Jurnal Pengembangan Teknologi Informasi Dan Ilmu Komputer*, *2*(12), 6269–6276.