

**IMPLEMENTASI DAN PENGUJIAN *REST API BACKEND* PADA
SISTEM PENJUALAN CITRA CAKE MENGGUNAKAN *POSTMAN*
(Studi Kasus: UMKM Citra Cake Sidamulih)**

SKRIPSI

**Diajukan untuk memenuhi sebagian persyaratan mendapatkan gelar Strata
Satu Program Studi Informatika**



**CHOLIFAH KUSUMA DEWI
22EO10010**

**PROGRAM STUDI INFORMATIKA
FAKULTAS MATEMATIKA DAN ILMU KOMPUTER
UNIVERSITAS NAHDLATUL ULAMA AL GHAZALI
CILACAP
2026**

**IMPLEMENTASI DAN PENGUJIAN *REST API BACKEND* PADA
SISTEM PENJUALAN CITRA CAKE MENGGUNAKAN *POSTMAN*
(Studi Kasus: UMKM Citra Cake Sidamulih)**

SKRIPSI

**Diajukan untuk memenuhi sebagian persyaratan mendapatkan gelar Strata
Satu Program Studi Informatika**



**CHOLIFAH KUSUMA DEWI
22EO10010**

**PROGRAM STUDI INFORMATIKA
FAKULTAS MATEMATIKA DAN ILMU KOMPUTER
UNIVERSITAS NAHDLATUL ULAMA AL GHAZALI
CILACAP
2026**

HALAMAN PERNYATAAN KEASLIAN SKRIPSI

Dengan ini saya menyatakan bahwa penulisan skripsi dengan judul "Implementasi dan Pengujian *Rest Api Backend* pada Sistem Penjualan Citra Cake menggunakan *Postman* (Studi Kasus: UMKM Citra Cake Sidamulih)" adalah hasil karya saya dengan arahan dari pembimbing dan belum diajukan kepada pihak manapun. Sumber informasi yang dikutip dalam skripsi ini telah dicantumkan dalam Daftar Pustaka.

Demikian pernyataan ini dibuat dengan sebenarnya. Apabila di kemudian hari terdapat ketidaksesuaian dalam pernyataan ini, maka saya bersedia menerima sanksi sesuai dengan peraturan yang berlaku.

Cilacap, 14 Agustus 2026



Cholifah Kusuma Dewi
22EO10010

LEMBAR PENGESAHAN

Skripsi Saudari

Nama : Cholifah Kusuma Dewi
NIM : 22EO10010
Fakultas / Prodi : Fakultas MIKOM / Informatika
Judul : Implementasi dan Pengujian *Rest Api Backend* pada Sistem Penjualan Citra Cake menggunakan *Postman* (Studi Kasus: UMKM Citra Cake Sidamulih)

Telah disidangkan oleh Dewan Penguji Fakultas Matematika dan Ilmu Komputer Universitas Nahdlatul Ulama Al Ghazali (UNUGHA) Cilacap pada hari / tanggal:

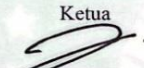
Rabu, 12 Agustus 2026

Dan dapat diterima sebagai pemenuhan tugas akhir mahasiswa Program Strata Satu Program Studi Informatika (S.Kom.) Fakultas Matematika dan Ilmu Komputer (FMikom) pada Universitas Nahdlatul Ulama Al Ghazali (UNUGHA) Cilacap.

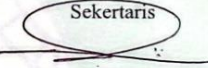
Cilacap, 14 Agustus 2026

Dewan Sidang

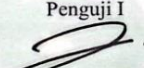
Ketua


Mochamad F. A. Aziz Zein, S.Si., M.Kom.
NIDN. 2125098601

Sekretaris


Tri Anggoro, M.Kom
NIDN. 0601128103

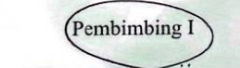
Penguji I


Mochamad F. A. Aziz Zein, S.Si., M.Kom.
NIDN. 2125098601

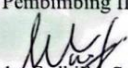
Penguji II


Eri Nur Shofi'i, S.H., M.A.
NIDN. 41230714190

Pembimbing I


Tri Anggoro, M.Kom
NIDN. 0601128103

Pembimbing II


Widya Rizky Fadhillah, S.Pd., M.Sc.
NUPTK. 6943774675230282

Mengetahui,

Dekan Fakultas Matematika dan Ilmu Komputer


Mochamad F. A. Aziz Zein, S.Si, M.Kom.
NIDN. 2125098601

iv

HALAMAN NOTA KONSULTAN

M.T.Abdul Aziz Zein, S.Si, M.Kom

Dosen Fakultas Matematika dan Ilmu Komputer
Universitas Nahdlatul Ulama Al Ghazali Cilacap

NOTA KONSULTAN

Hal : Skripsi Saudara/i Cholifah Kusuma Dewi
Lampiran : -

Kepada

Yth. Dekan Fakultas Matematika dan Ilmu Komputer
Universitas Nahdlatul Ulama Al Ghazali Cilacap
di Cilacap

Assalamu'alaikum Wr. Wb.

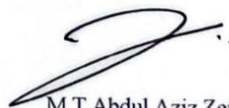
Setelah membaca, memeriksa dan melakukan perbaikan seperlunya maka skripsi saudara/i:

Nama : Cholifah Kusuma Dewi
NIM : 22EO10010
Prodi : Informatika
Judul : Implementasi dan Pengujian *Rest Api Backend* pada Sistem
Penjualan Citra Cake menggunakan *Postman* (Studi Kasus:
UMKM Citra Cake Sidamulih)

Dapat diajukan ke Fakultas Matematika dan Ilmu Komputer, Universitas Nahdlatul Ulama Al Ghazali Cilacap untuk memenuhi syarat memperoleh gelar Strata Satu (S1).

Wassalamu'alaikum Wr. Wb.

Cilacap, 14 Agustus 2026
Konsultan



M.T.Abdul Aziz Zein, S.Si, M.Kom
NIDN. 2125098601

NOTA PEMBIMBING

Kepada Yth :
Fakultas Matematika dan Komputer (FMikom)
UNUGHA Cilacap
di Cilacap

Cilacap, 06 Agustus 2026

Assalamu'alaikum Wr. Wb.

Setelah melakukan bimbingan, telah, arahan dan koreksi tahap penulisan skripsi saudara:

Nama : Cholifah Kusuma Dewi
NIM : 22EO10010
Fakultas : Matematika dan Ilmu Komputer UNUGHA Cilacap
Prodi : Informatika
Judul : Implementasi dan Pengujian *Rest Api Backend* pada Sistem Penjualan Citra Cake menggunakan *Postman* (Studi Kasus: UMKM Citra Cake Sidamulih)

Kami berpendapat bahwa skripsi tersebut sudah dapat diajukan ke sidang skripsi. Bersamaan ini kami kirimkan skripsi tersebut, semoga dapat segera disidangkan.

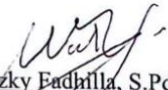
Atas perhatiannya kami ucapkan terima kasih.
Wassalamu'alaikum Wr. Wb.

Mengetahui,

Pembimbing I

Pembimbing II


Tri Anggoro, M.Kom.
NIDN. 0601128103


Widya Rizky Fadhillah, S.Pd., M.Sc.
NUPTK. 6943774675230282

HALAMAN PERSEMBAHAN

Segala puji bagi Allah SWT, Rabb semesta alam yang senantiasa memberikan karunia sehingga penulis mampu menyelesaikan penulisan skripsi ini. Karya ini saya persembahkan kepada :

1. Orang tua (Bapak Satun Saputro dan Ibu Elis Lestari Wahyuningsih) yang tercinta selalu mendampingi, mendukung, dan mendoakan supaya sukses dalam mengerjakan skripsi. Terima kasih mungkin tanpa kalian saya belum tentu sekuat sampai sekarang.
2. Kepada dosen pembimbing Bapak Tri Anggoro, M.kom dan Ibu Widya Rizky Fadhillah, S.Pd., M.Sc yang sudah membimbing saya dalam mengerjakan skripsi terima kasih atas doa dan dukungannya.
3. Rekan Saya Ika Nur Hidayati terima kasih sudah selalu saling mendoakan dan mendukung satu sama lain dalam mengerjakan skripsi.
4. Terima kasih kepada keluarga saya yang selalu memberikan doa dan dukungan dalam setiap langkahnya.
5. Terima kasih kepada teman-temannya saya FMIKOM Angkatan 2022 yang selalu memberikan dukungan, doa, dan motivasi.
6. Terima kasih kepada program studi INFORMATIKA UNUGHA CILACAP yang saya banggakan.

KATA PENGANTAR

Puji syukur kepada Allah SWT atas segala limpahan nikmat dan karuniaNya, sehingga penulis dapat menyelesaikan penulisan skripsi. Sholawat dan salam senantiasa penulis haturkan kepada Nabi Muhammad SAW sebagai pembimbing seluruh umat manusia.

Skripsi ini tidak akan selesai tanpa adanya bantuan dari banyak pihak, karena itu penulis menyampaikan terima kasih kepada :

1. Bapak Mochamad T. A. Aziz Zein, S.Si., M.Kom. , Dekan FMIKOM UNUGHA
2. Bapak Tri Anggoro, M.Kom, pembimbing I yang telah dengan sabar memberikan bimbingan dan petunjuk dalam menyelesaikan skripsi ini.
3. Ibu Widya Rizky Fadhilla, S.Pd., M.Sc, pembimbing II terima kasih yang telah dengan memberikan bimbingan dan arahan dalam menyelesaikan skripsi ini.
4. Kepada UMKM Citra Cake terima kasih yang sudah mendukung selama mengerjakan skripsi.
5. Bapak-Ibu dosen Program Studi Informatika FMIKOM UNUGHA CILACAP.
6. Semua pihak yang tidak dapat kami sebutkan satu persatu.

Semoga Allah SWT membalas jerih payah dan pengorbanan yang telah diberikan dengan balasan yang lebih baik. Amiin. Penulis berharap semoga karya kecil ini bermanfaat bagi pembaca.

Cilacap, 14 Agustus 2026



Cholifah Kusuma Dewi

ABSTRAK

Cholifah Kusuma Dewi. Implementasi dan Pengujian *Rest Api Backend* pada Sistem Penjualan Citra Cake menggunakan *Postman* (Studi Kasus: UMKM Citra Cake Sidamulih). Dibimbing oleh Tri Anggoro, M.Kom dan Widya Rizky Fadhilla, S.Pd., M.Sc.

Kemajuan teknologi informasi mendorong pelaku Usaha Mikro, Kecil, dan Menengah (UMKM) untuk memanfaatkan sistem digital guna mendukung operasional bisnis. UMKM Citra Cake merupakan usaha yang bergerak di bidang penjualan dan pemesanan kue, namun pengelolaan data produk, stok, dan pesanan sebelumnya masih dilakukan secara manual sehingga berpotensi menimbulkan kesalahan pencatatan, keterlambatan layanan, dan kesulitan pengelolaan transaksi. Penelitian ini menghasilkan *REST API* backend untuk mendukung pengelolaan data pada Sistem Penjualan Citra Cake.

REST API backend diimplementasikan menggunakan *Node.js* dan *Express.js* sebagai lingkungan pengembangan *server*, *PostgreSQL* sebagai basis data, serta *Prisma ORM* sebagai penghubung aplikasi dengan basis data. Proses pengembangan menggunakan pendekatan *Agile Development* melalui tahap *plan, design, develop, test, deploy, dan review*. *REST API* yang dihasilkan menyediakan *endpoint* untuk autentikasi, profil, produk, kategori, keranjang, dan pemesanan.

Pengujian dilakukan menggunakan *Postman* dengan pendekatan *Black Box Testing* pada setiap *endpoint* melalui skenario *Single Track* dan *Multi Track*. Parameter yang diamati meliputi *Response Time, Status Code, dan success rate*. Hasil pengujian menunjukkan bahwa seluruh *endpoint* yang diuji berhasil diproses sesuai fungsi, memperoleh *Status Code* yang sesuai, dan mencapai *success rate* 100% pada pengujian *Single Track* maupun *Multi Track*. Dengan demikian, *REST API backend* yang diimplementasikan telah memenuhi kebutuhan fungsional backend pada Sistem Penjualan Citra Cake dan dapat digunakan sebagai layanan komunikasi antara frontend dan basis data.

Kata kunci: *REST API backend, Node.js, PostgreSQL, Prisma ORM, Postman, sistem penjualan berbasis web.*

ABSTRACT

Cholifah Kusuma Dewi. *Implementation and Testing of a Rest Api Backend for the Citra Cake Sales System Using Postman (Case Study: Citra Cake MSME, Sidamulih). Supervised by Tri Anggoro, M.Kom. and Widya Rizky Fadhilla, S.Pd., M.Sc.*

Information technology encourages Micro, Small, and Medium Enterprises (MSMEs) to adopt digital systems to support business operations. Citra Cake is a cake sales and ordering business whose product, inventory, and order management was previously handled manually, creating potential recording errors, service delays, and difficulties in transaction management. This research produced a REST API backend to support data management in the Citra Cake Sales System.

The backend was implemented using Node.js and Express.js as the server environment, PostgreSQL as the database, and Prisma ORM as the interface between the application and database. Development followed an Agile Development approach consisting of plan, design, develop, test, deploy, and review stages. The resulting REST API provides endpoints for authentication, profiles, products, categories, cart, and orders.

Testing was conducted using Postman with a Black Box Testing approach on each endpoint through Single Track and Multi Track scenarios. The observed parameters were Response Time, Status Code, and success rate. The results show that all tested endpoints were processed according to their functions, returned appropriate Status Codes, and achieved a 100% success rate in both Single Track and Multi Track testing. Therefore, the implemented REST API backend fulfilled the functional backend requirements of the Citra Cake Sales System and can serve as a communication service between the frontend and database.

Keywords: *REST API backend, Node.js, PostgreSQL, Prisma ORM, Postman, web-based sales system.*

DAFTAR ISI

HALAMAN PERNYATAAN KEASLIAN SKRIPSI	Error! Bookmark not defined.
LEMBAR PENGESAHAN	Error! Bookmark not defined.
HALAMAN NOTA KONSULTAN.....	iv
HALAMAN NOTA PEMBIMBING.....	vi
HALAMAN MOTO	vi
HALAMAN PERSEMBAHAN	vii
KATA PENGANTAR	Error! Bookmark not defined.
ABSTRAK.....	ix
<i>ABSTRACT</i>	x
DAFTAR ISI.....	xi
DAFTAR GAMBAR	xiii
DAFTAR TABEL.....	xiv
BAB I PENDAHULUAN	1
A. Latar Belakang	1
B. Rumusan Masalah.....	3
C. Batasan Masalah	3
D. Tujuan Penelitian	4
E. Manfaat Penelitian	4
BAB II TINJAUAN PUSTAKA	5
A. Penelitian Terkait	5
B. Landasan Teori.....	10
BAB III METODOLOGI.....	12
A. Waktu dan Tempat Penelitian.....	12
B. Alat dan Bahan.....	12
C. Prosedur Penelitian	13
D. Analisis Data.....	23
E. Jadwal Penelitian	24
BAB IV HASIL DAN PEMBAHASAN.....	25
A. Hasil	25
B. Pembahasan.....	39

BAB V KESIMPULAN.....	40
A. Kesimpulan	40
B. Saran	40
LAMPIRAN.....	44

DAFTAR GAMBAR

Gambar 1 <i>Mind Map</i>	10
Gambar 2 <i>Metode Agile Development</i>	11
Gambar 3 <i>Flowchart</i> Penelitian	13
Gambar 4 <i>Use Case Diagram</i>	14
Gambar 5 <i>Activity Diagram</i>	15
Gambar 6 <i>Sequence Diagram</i>	16
Gambar 7 <i>Class Diagram</i>	17
Gambar 8 Jadwal Penelitian.....	24
Gambar 9 <i>Database</i>	26
Gambar 10 <i>Schema Prisma</i>	27
Gambar 11 <i>Clouldinary</i>	28
Gambar 12 <i>Register</i>	30
Gambar 13 <i>Products</i>	31
Gambar 14 <i>Cart</i>	31
Gambar 15 <i>Checkout</i>	32
Gambar 16 <i>Pesanan</i>	32
Gambar 17 <i>Review</i>	33
Gambar 18 <i>Tampilan Register</i>	36
Gambar 19 <i>Tampilan Login</i>	36
Gambar 20 <i>Tampilan Produk</i>	39
Gambar 21 <i>Register</i>	44
Gambar 22 <i>User</i>	44
Gambar 23 <i>User</i>	44
Gambar 24 <i>Product</i>	45
Gambar 25 <i>Options</i>	45
Gambar 26 <i>Layer Cream</i>	45
Gambar 27 <i>Custom Cake</i>	46
Gambar 28 <i>Options Custom Cake</i>	46
Gambar 29 <i>Warna Cream</i>	46
Gambar 30 <i>Best Seller</i>	47
Gambar 31 <i>Newest</i>	47
Gambar 32 <i>Sukses Category</i>	48
Gambar 33 <i>Owner</i>	48
Gambar 34 <i>Sukses data</i>	48
Gambar 35 <i>Checkout</i>	49
Gambar 36 <i>My Order</i>	49
Gambar 37 <i>Sukses Order</i>	49
Gambar 38 <i>Payment</i>	50
Gambar 39 <i>Pilihan Payment</i>	50
Gambar 40 <i>Admin</i>	50

DAFTAR TABEL

Tabel 1 Penelitian Terkait	5
Tabel 2 Alat dan Bahan	12
Tabel 3 Pengujian <i>API</i> Autentikasi	18
Tabel 4 Pengujian <i>API Profile</i>	19
Tabel 5 Pengujian <i>API</i> Produk	19
Tabel 6 Pengujian <i>API</i> Kategori	20
Tabel 7 Pengujian <i>API Cart</i>	20
Tabel 8 Pengujian <i>API Order</i>	21
Tabel 9 Pengujian Respons <i>REST API</i> Menggunakan <i>Postman</i>	21
Tabel 10 Implementasi <i>Endpoint REST API</i>	28
Tabel 11 Pengujian <i>Single Track</i>	34
Tabel 12 Pengujian <i>Multi Track</i>	37

BAB I PENDAHULUAN

A. Latar Belakang

Kemajuan teknologi informasi telah menghadirkan perubahan signifikan di berbagai sektor, termasuk usaha mikro, kecil, dan menengah (UMKM). Digitalisasi dalam proses bisnis merupakan salah satu strategi yang dapat diterapkan untuk meningkatkan efisiensi operasional, memperluas jangkauan pemasaran, serta meningkatkan kualitas layanan kepada konsumen[1]. Salah satu implementasi dari digitalisasi ini adalah pemanfaatan sistem penjualan berbasis *web* yang mampu mengelola data produk, pelanggan, stok, dan transaksi secara terintegrasi[2]. UMKM Citra Cake Sidamulih adalah bisnis yang berfokus pada penjualan produk kue. Dalam operasionalnya, pengelolaan data produk, stok, dan pemesanan memerlukan sistem yang dapat mengolah dan menyajikan data secara cepat dan akurat[3]. Seiring dengan pertumbuhan kebutuhan transaksi dan pengelolaan informasi, diperlukan sebuah sistem *backend* yang bisa berfungsi sebagai pusat pengelolaan data serta mendukung interaksi antara basis data dengan antarmuka pengguna (*frontend*)[4]. Perancangan sistem penjualan UMKM Citra Cake Sidamulih dilaksanakan secara kolaboratif dengan pembagian tugas antara sisi *frontend* dan *backend*. Bagian *frontend* dirancang oleh rekan peneliti menggunakan *React.js* yang berfungsi sebagai antarmuka pengguna (*client*) yang berinteraksi dengan *REST API backend*. Sementara itu, penelitian ini berfokus pada bagian *backend* yang menggunakan *Node.js* dan *Express.js* sebagai penyedia layanan *REST API*. Pembagian tugas ini memungkinkan *frontend* dan *backend* untuk dirancang secara terpisah namun tetap terintegrasi melalui pertukaran data menggunakan *REST API*. Oleh karena itu, penerapan *REST API (Representational State Transfer Application Programming Interface)* menjadi solusi yang ideal karena bersifat ringan, fleksibel, dan mudah diintegrasikan dengan berbagai *platform*[5].

REST API merupakan gaya arsitektur layanan yang menggunakan *HTTP* dan *URI* untuk menyediakan akses terhadap sumber daya aplikasi secara terstruktur. Penerapan aturan *REST* pada *Web API* membantu memperjelas penggunaan resource, metode *HTTP*, dan *Status Code* sehingga antarmuka layanan lebih mudah dipahami dan digunakan. Penelitian empiris menunjukkan bahwa kepatuhan terhadap aturan desain *REST* berpengaruh terhadap keterpahaman *Web API*[6]. Dalam penelitian ini, *REST API* digunakan sebagai lapisan layanan *backend* karena memungkinkan *frontend* berkomunikasi dengan data dan proses bisnis melalui *endpoint* yang terpisah dari antarmuka pengguna. *Backend* dirancang menggunakan *Node.js* dan *Express.js*, *PostgreSQL* sebagai sistem manajemen basis data, serta *Prisma ORM* sebagai penghubung aplikasi dengan database.

Pengujian *REST API* merupakan bagian penting untuk memastikan layanan *backend* menghasilkan respons sesuai spesifikasi. Penelitian mengenai pengujian *RESTful API* menunjukkan bahwa pengujian *black-box* dapat digunakan untuk memeriksa perilaku *API* melalui *input* dan *output* tanpa bergantung pada kode sumber[7], sedangkan pengujian berkelanjutan dapat digunakan untuk memantau *correctness* dan *reliability API*[8]. Salah satu alat yang digunakan dalam penelitian ini adalah

Postman karena memungkinkan pengiriman *request* ke setiap *endpoint* dan pemeriksaan *response* secara langsung. Penelitian Gonzaga dan Nawaningtyas menunjukkan bahwa REST API dapat diimplementasikan sebagai *backend service* untuk menyediakan layanan pengelolaan data melalui *endpoint*. Penelitian tersebut melakukan validasi fungsional menggunakan *Postman* dan menunjukkan bahwa seluruh *endpoint* untuk operasi *Create, Read, Update, dan Delete* (CRUD) berhasil diimplementasikan dan berfungsi sesuai dengan spesifikasi yang diharapkan[9]. Hasil tersebut menjadi salah satu dasar dalam penelitian ini untuk menggunakan *Postman* sebagai alat pengujian fungsional terhadap *endpoint REST API backend* pada sistem penjualan Citra Cake Sidamulih.

Penggunaan *Postman* dalam penelitian ini ditujukan untuk menguji *endpoint-endpoint* yang terkait dengan autentikasi pengguna, pengelolaan produk, kategori, profil pengguna, keranjang belanja, dan pemesanan. Pengujian dilaksanakan dengan memeriksa validitas permintaan dan respons, *Status Code HTTP*, serta keberhasilan setiap fungsi *endpoint* yang telah dirancang. Dengan demikian, kualitas dan keandalan *REST API backend* dapat dinilai sebelum sistem dioperasikan. Penelitian Thooriqoh, Mulyo, dan Rakhmadi melakukan pengujian *REST API* menggunakan *Postman* pada transaksi *file upload*. Penelitian tersebut membandingkan penggunaan *Postman* dan *Cucumber* dengan memperhatikan penggunaan sumber daya, seperti prosesor, memori, I/O, penyimpanan, dan *bandwidth*. Hasil penelitian menunjukkan bahwa *Postman* memiliki efisiensi yang lebih baik pada sebagian besar parameter penggunaan sumber daya yang diuji [10]. Hasil tersebut mendukung penggunaan *Postman* sebagai alat pengujian *REST API* pada penelitian ini.

Berdasarkan penelitian terdahulu, pengujian *REST API* diperlukan untuk memastikan *endpoint* memberikan respons sesuai dengan fungsi yang dirancang. Golmohammadi, Zhang, dan Arcuri melalui kajian terhadap 92 penelitian tentang pengujian *RESTful API* menunjukkan bahwa pengujian *API* penting untuk memverifikasi layanan yang berkomunikasi melalui jaringan dan berinteraksi dengan layanan eksternal seperti basis data [5]. Berdasarkan penelitian tersebut, penelitian ini melanjutkan penerapan *REST API* pada sistem penjualan UMKM Citra Cake Sidamulih dengan menguji *endpoint* menggunakan *Postman*. Pengujian dilakukan melalui skenario *Single Track* dengan satu kali *request* pada setiap *endpoint* dan *Multi Track* dengan lima kali *request* pada *endpoint* yang diuji. Parameter yang diamati meliputi *Response Time, Status Code, dan Success Rate*. Penelitian Poth, Rrjolli, dan Arcuri membahas pengujian *black-box* pada *REST API* dengan menggunakan spesifikasi *OpenAPI* sebagai dasar pembentukan dan evaluasi *test case*. Penelitian tersebut menunjukkan bahwa pengujian *REST API* diperlukan untuk memverifikasi kesesuaian layanan terhadap kebutuhan dan spesifikasi yang telah ditentukan [11]. Hal tersebut sejalan dengan penelitian ini yang melakukan pengujian terhadap setiap *endpoint* berdasarkan *request dan response* yang dihasilkan. Penelitian Yanuardi, Aminudin, dan Faiqurahman melakukan evaluasi performa *RESTful API* menggunakan *Postman* dan *JMeter* dengan parameter waktu respons, ukuran respons, penggunaan CPU, *throughput*, dan waktu muat [12]. Penelitian tersebut menunjukkan bahwa waktu respons dapat digunakan sebagai salah satu parameter untuk mengevaluasi kinerja layanan *API*. Dalam penelitian ini, *Response Time*

digunakan untuk mencatat waktu yang diperlukan *backend* dalam memberikan respons terhadap *request*, sedangkan pengujian tidak diarahkan sebagai pengujian beban secara formal.

Beberapa penelitian sebelumnya menunjukkan bahwa penerapan *REST API* dan pengujian *API* dapat membantu memastikan layanan *backend* berfungsi sesuai kebutuhan. Karlsson et al. melakukan penelitian mengenai perilaku *RESTful API* pada lingkungan industri dan menjelaskan bahwa *REST API* merupakan pendekatan yang umum digunakan untuk menyediakan fungsi sistem melalui *Web API*. Penelitian tersebut menganalisis perilaku *RESTful API* berdasarkan interaksi antara *client* dan layanan yang disediakan *API* [13]. Penelitian tersebut mendukung penelitian ini dalam melakukan pengujian terhadap *endpoint* berdasarkan *request* yang diberikan dan *response* yang dihasilkan. Kajian Golmohammadi, Zhang, dan Arcuri juga menunjukkan pentingnya pengujian *RESTful API* untuk memverifikasi perilaku layanan [5]. Penelitian ini menerapkan dan menguji *REST API backend* pada sistem penjualan UMKM Citra Cake Sidamulih menggunakan *Postman* dengan fokus pada validasi fungsi *endpoint* serta pengamatan *Response Time*, *Status Code*, dan *Success Rate* melalui *Single Track* dan *Multi Track*.

Berdasarkan uraian tersebut, penelitian ini berjudul “Implementasi dan Pengujian *Rest Api Backend* pada Sistem Penjualan Citra Cake menggunakan *Postman* (Studi Kasus: UMKM Citra Cake Sidamulih)”. Penelitian berfokus pada implementasi *REST API backend* dan pengujian setiap *endpoint* yang menjadi objek pengujian menggunakan *Postman* sehingga fungsi layanan dapat divalidasi secara terukur.

B. Rumusan Masalah

1. Bagaimana mengimplementasikan *REST API backend* pada sistem penjualan Citra Cake menggunakan *Node.js*, *PostgreSQL*, dan *Prisma ORM*?
2. Bagaimana hasil pengujian *endpoint REST API backend* pada sistem penjualan Citra Cake menggunakan *Postman* berdasarkan parameter *Response Time*, *Status Code*, dan tingkat keberhasilan?

C. Batasan Masalah

Penelitian berfokus pada implementasi *REST API backend* pada Sistem Penjualan Citra Cake Sidamulih menggunakan *Node.js*, *Express.js*, *PostgreSQL*, dan *Prisma ORM*.

- a. *Endpoint* yang diuji mencakup autentikasi, profil pengguna, produk, kategori, keranjang belanja, dan pemesanan.
- b. Pengujian *REST API* dilakukan menggunakan *Postman* dengan pendekatan *Black Box Testing* melalui skenario *Single Track* dan *Multi Track*.
- c. Parameter pengujian yang dianalisis meliputi *Response Time*, *Status Code*, dan *Success Rate*.
- d. Penelitian tidak membahas perancangan *frontend* karena bagian *frontend* dikerjakan secara terpisah oleh rekan dalam penelitian kolaboratif.
- e. Penelitian tidak melakukan *security testing* secara khusus karena fokus penelitian adalah validasi fungsi *endpoint* melalui *request*, *response*, dan *Status Code*. Pengujian keamanan memerlukan skenario ancaman dan teknik pengujian khusus. Penelitian Syahroni et al. menunjukkan bahwa pengujian keamanan *backend* dapat dilakukan dengan pendekatan dan skenario khusus [14].

- f. *Response Time* dicatat sebagai informasi mengenai waktu respons layanan dan bukan merupakan *performance/load testing* formal karena pengujian beban memerlukan rancangan beban, *throughput*, *concurrent users*, dan alat khusus.
- g. Studi kasus penelitian dilakukan pada UMKM Citra Cake Sidamulih dan *backend* yang telah diimplementasikan dideploy pada *Virtual Private Server (VPS)*.

D. Tujuan Penelitian

1. Mengimplementasikan *REST API backend* pada Sistem Penjualan Citra Cake Sidamulih menggunakan *Node.js*, *Express.js*, *PostgreSQL*, dan *Prisma ORM*.
2. Menguji *endpoint REST API backend* menggunakan *Postman* melalui skenario *Single Track* dan *Multi Track* berdasarkan *Response Time*, *Status Code*, dan *Success Rate*.

E. Manfaat Penelitian

Penelitian ini diharapkan dapat memberikan manfaat sebagai berikut:

1. Manfaat Bagi Peneliti: Penelitian ini dapat meningkatkan pengetahuan dan keterampilan peneliti dalam mengimplementasikan serta menguji *REST API backend* menggunakan *Node.js*, *PostgreSQL*, *Prisma ORM*, dan *Postman*, serta menerapkan ilmu yang diperoleh selama perkuliahan pada permasalahan nyata.
2. Manfaat Akademik: Penelitian ini diharapkan dapat menjadi referensi dan sumber literatur bagi mahasiswa dan peneliti di bidang rekayasa perangkat lunak, khususnya mengenai implementasi *REST API backend* dan pengujian *API* menggunakan *Postman*.
3. Manfaat Bagi UMKM Citra Cake Sidamulih: Penelitian ini diharapkan dapat membantu UMKM Citra Cake Sidamulih dalam mendukung pengelolaan data produk operasional menjadi lebih efektif dan efisien.

BAB II TINJAUAN PUSTAKA

A. Penelitian Terkait

Digunakan sebagai dasar ilmiah untuk menunjukkan perkembangan penelitian *REST API*, pengujian *endpoint*, dan teknologi *backend* yang relevan. Referensi dipilih terutama dari penelitian yang terbit dalam lima tahun terakhir agar posisi penelitian ini dapat dibandingkan dengan penelitian terdahulu. Penelitian terkait disajikan untuk menunjukkan persamaan, perbedaan, dan posisi penelitian pada implementasi serta pengujian *REST API backend* sistem penjualan UMKM.

Tabel 1 Penelitian Terkait

No	Nama	Judul
(1)	(2)	(3)
1	(Syahroni et al., 2024)	Judul: Uji Keamanan <i>Back End</i> Aplikasi Berbasis <i>Website</i> Menggunakan Metode <i>Black Box Testing</i> Metode: <i>Black Box Testing</i> . Software: <i>Postman</i> , <i>JWT (JSON Web Token)</i> . Hasil Penelitian: Hasil penelitian <i>authentication</i> berjalan baik, karena <i>API</i> tidak dapat diakses tanpa <i>JWT</i> dan akan memberikan 401 <i>unauthorized</i> .
2	(Jati et al., 2025)	Judul: Efisiensi Penggunaan <i>Raw Query</i> dan <i>ORM</i> pada Ekosistem <i>Node.js</i> Metode: <i>Eksperimen komparatif</i> Software: <i>Node.js</i> , <i>Express.js</i> , <i>PostgreSQL</i> , <i>Prisma ORM</i> , <i>node-pg</i> , <i>Grafana K6</i> Hasil Penelitian: Hasil penelitian menunjukkan bahwa terdapat perbedaan performa yang signifikan antara <i>Raw Query</i> dan <i>ORM</i> . Pada operasi <i>read</i> , <i>Raw Query</i> menghasilkan median <i>response time</i> sebesar 3,48 ms, sedangkan <i>ORM</i> 6,26 ms.

3	(Rangga, Guntara & Azkari, 2023)	Judul: Implementasi dan Pengujian <i>REST API</i> Sistem Reservasi Ruang Rapat dengan Metode <i>Black Box Testing</i> (Studi Kasus: PT Lizzie Parra Kreasi) Metode: <i>Software Development Life Cycle (SDLC)</i>. Software: <i>Node.js, Postman, REST API, Database Management System</i>. Hasil Penelitian: Hasil penelitian pengujian dengan metode <i>black box testing</i>, yang menunjukkan bahwa semua fungsi setiap <i>endpoint API</i> berjalan sesuai dengan output yang diharapkan. Melalui metode <i>HTTP</i>, seluruh <i>endpoint</i> dapat berjalan dengan baik, mulai dari perolehan data dengan metode <i>GET</i>, pembaharuan data dengan metode <i>PUT</i>, penambahan data dengan metode <i>POST</i>, dan penghapusan data dengan metode <i>DELETE</i>.
4	(Muhaimma & Paputangan, 2024)	Judul: Pengembangan <i>Backend Server</i> Berbasis Arsitektur <i>REST API</i> pada Sistem Transfer Dompot Digital Metode: <i>Extreme Programming (XP)</i> Software: <i>NestJS, Node.js, MySQL, Postman, Visual Studio Code, Swagger</i>. Hasil Penelitian: Hasil penelitian berhasil mengimplementasikan pemindahan saldo antar dompet digital. Semua <i>API</i> telah berhasil diimplementasi dan diuji secara berkala mulai kegunaan, <i>error handling</i>, dan lain sebagainya.
5	(Sulaiman et al., 2023)	Judul: Perancangan Aplikasi Todo Menggunakan <i>Node.js</i> dan <i>REST API</i> Metode: <i>Agile</i>. Software: <i>Node.js, Express.js, REST API, Postman</i>. Hasil Penelitian: Hasil penelitian menunjukkan bahwa penerapan teknologi modern seperti <i>node.js</i> dan metodologi <i>agile</i> dalam pengembangan aplikasi dapat menghasilkan solusi yang responsif dan efektif bagi pengguna.

6 (Arief et al., 2025)	Judul: Pengembangan Sistem Informasi Transaksi Penjualan UMKM menggunakan Metode <i>Rapid Application Development</i> Metode: <i>Rapid Application Development (RAD)</i> Software: <i>ExpressJS, EJS (Embedded JavaScript), SQLite, Model-View-Controller (MVC)</i> Hasil Penelitian: Hasil penelitian berhasil mengembangkan sistem informasi transaksi penjualan berbasis <i>web</i> yang sesuai dengan kebutuhan operasional UMKM Uwais Telur. Sistem menyediakan fitur pencatatan transaksi, pengelolaan data pelanggan, monitoring pembayaran, pencetakan nota, manajemen stok, serta pencadangan data ke format Excel. Hasil <i>Alpha Testing</i> menunjukkan seluruh fitur berjalan dengan baik dan sesuai kebutuhan sistem. Hasil <i>Beta Testing</i> juga menunjukkan bahwa pengguna merasa sistem lebih sederhana, ringan, mudah digunakan, serta lebih sesuai dibandingkan aplikasi yang di gunakan sebelumnya.
7 (Naufal et al., 2022)	Judul: Implementasi <i>REST API</i> untuk Fitur Rencana Strategis Program Pada SIMPEDIA Metode: Pengembangan Sistem. Software: <i>Frame Work Phalcon, PHP, Javascript, Postman.</i> Hasil Penelitian: Hasil implementasi <i>REST API</i> berhasil mempermudah pertukaran data antara <i>client</i> dan <i>server</i>, meningkatkan efisiensi sistem, serta menghasilkan seluruh <i>endpoint CRUD</i>

		yang diuji dengan status sukses. <i>Output</i> data menggunakan format <i>JSON</i> sehingga lebih mudah diproses oleh <i>client</i> .
8	(Hasanudin, 2022)	Judul: Rancangan Bangun <i>REST API</i> Aplikasi sebagai Upaya Mempermudah Pelayanan Donasi Kemanusiaan Metode: <i>SCRUM</i> Software: <i>Node.js, ExpressJS, PostgreSQL, Sequelize (ORM), Postman</i>. Hasil Penelitian: Hasil pengujian merancang dan membangun sistem berbasis <i>REST API</i> dengan arsitektur <i>REST</i> pada sisi <i>backend</i> yang dapat diimplementasikan pada aplikasi <i>website</i> dan <i>mobile (Android)</i> untuk layanan donasi <i>online</i>. Sistem memudahkan proses transaksi donasi, pertukaran data menggunakan format <i>JSON</i>, serta mendukung pengelolaan dan verifikasi donasi secara lebih efisien.
9	(Listyano et al., 2024)	Judul: Optimasi Aplikasi Toko Bangunan Melalui Integrasi <i>Rest API</i> dan <i>WhatsApp Gateway</i> Metode: : Analisis Kebutuhan Sistem. Software: <i>REST API, WhatsApp Gateway, Database SQL, Aplikasi E-commerce</i>. Hasil Penelitian: Hasil Penelitian berhasil mengembangkan sistem berbasis <i>REST API</i> yang mengintegrasikan manajemen stok, pemesanan, pembayaran, dan pelacakan pengiriman. Hasil penelitian menunjukkan peningkatan efisiensi pengelolaan stok, peningkatan konektivitas antara sistem dan konsumen, serta kemudahan pemantauan pengiriman secara <i>real-time</i> sehingga operasional toko menjadi lebih efektif.

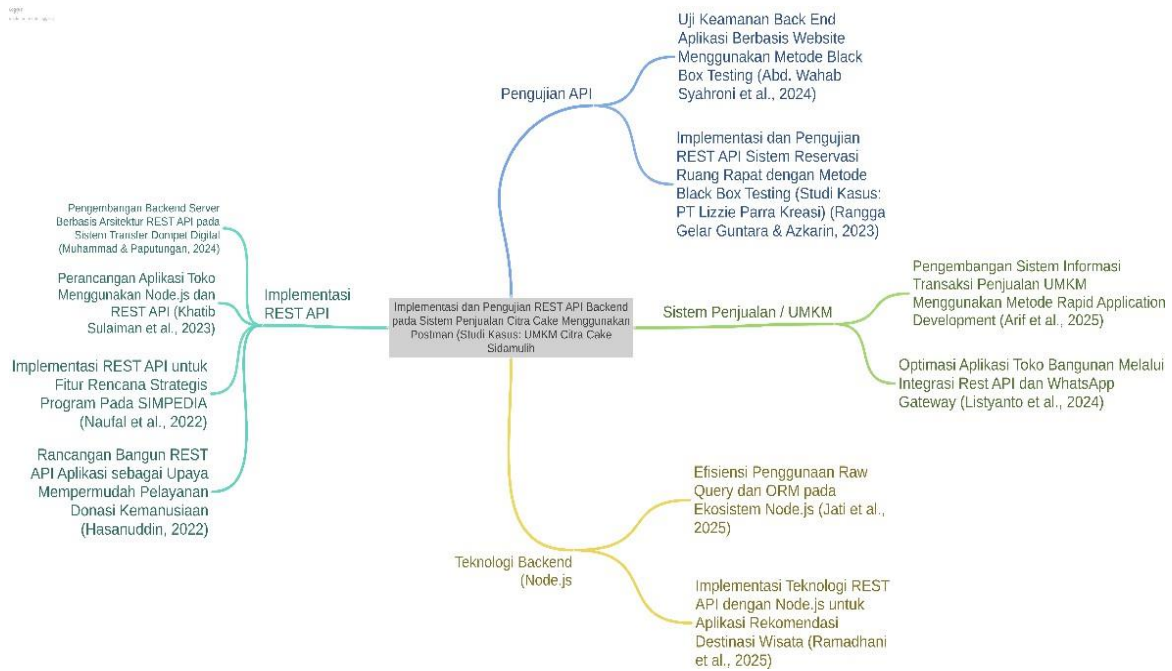
	Judul: Implementasi Teknologi <i>REST API</i> dengan <i>Node.js</i> untuk Aplikasi Rekomendasi Destinasi Wisata
	Metode: Observasi
	Software: <i>Node.js</i> , <i>REST API</i> , <i>Cloud Computing</i> , <i>Google Maps API</i> .
	Hasil Penelitian:
10 (Rama dhani et al., 2025)	Hasil penelitian <i>REST API</i> yang dibangun menggunakan <i>Node.js</i> berhasil menyediakan layanan data destinasi wisata secara cepat dan efektif. Implementasi ini menunjukkan bahwa <i>Node.js</i> memiliki performa yang baik untuk pengembangan <i>backend REST API</i> serta mendukung integrasi dengan berbagai sumber data dan peningkatan skalabilitas sistem.

Penelitian ini memiliki kebaruan dengan menggabungkan implementasi dan pengujian *REST API backend* pada sistem penjualan UMKM Citra Cake Sidamulih menggunakan teknologi modern. Secara objek penelitian, sistem yang dirancang berfokus pada pengelolaan produk, kategori, stok, keranjang belanja, *profile* pengguna, dan pemesanan kue yang mendukung kebutuhan operasional UMKM secara terintegrasi. Fitur tersebut membutuhkan pengelolaan data yang terstruktur dan komunikasi yang baik antara *frontend* dan *backend*.

Kebaruan penelitian juga terletak pada pengujian *endpoint REST API* menggunakan *Postman* untuk memverifikasi fungsi layanan *backend*. Pengujian dilakukan pada *endpoint* autentikasi, profil pengguna, produk, kategori, keranjang, dan pemesanan dengan memeriksa *request*, *response*, *status code*, *response time*, dan *success rate*. Pengujian *Single Track* dilakukan satu kali, sedangkan *Multi Track* dilakukan lima kali pada *endpoint* yang diuji. Hasil pengujian memberikan gambaran struktur mengenai keberhasilan fungsi layanan *backend* pada Sistem Penjualan Citra Cake Sidamulih.

Dengan pengukuran tersebut, penelitian dapat menunjukkan keberhasilan dan karakteristik repons layanan *REST API backend* berdasarkan hasil pengujian *Postman*.

Selain itu, penelitian ini tidak hanya berfokus pada implementasi *REST API backend*, tetapi juga pada evaluasi respons layanan yang dihasilkan melalui pengujian menggunakan *Postman*. Evaluasi dilakukan dengan mengukur *Response Time*, *Status Code*, dan tingkat keberhasilan *endpoint* pada setiap fitur yang tersedia. Hasil tersebut menjadi dasar untuk menilai apakah *endpoint* yang diuji telah memberikan respons sesuai dengan kebutuhan sistem penjualan UMKM Citra Cake Sidamulih. Dengan adanya pengukuran tersebut, penelitian ini dapat memberikan gambaran mengenai kualitas layanan *backend* yang diimplementasikan serta memastikan bahwa setiap *endpoint* mampu memberikan respons yang sesuai dengan kebutuhan sistem penjualan UMKM Citra Cake Sidamulih.



Gambar 1 Mind Map

B. Landasan Teori

1. Sistem Penjualan dan *Backend*

Sistem penjualan berbasis *web* merupakan sistem informasi yang mengelola data produk, pelanggan, stok, pemesanan, dan transaksi secara terintegrasi. Dalam penelitian ini, kebutuhan tersebut dipenuhi melalui pembagian fungsi antara *frontend* dan *backend*. *Backend* menjadi bagian *server-side* yang menangani logika bisnis, validasi, autentikasi, pengelolaan data, dan penyediaan layanan yang dapat diakses oleh *frontend* melalui *API*. Dengan demikian, *backend* menjadi pusat pemrosesan data yang menghubungkan proses bisnis sistem dengan basis data.

2. *REST API* sebagai Arsitektur Layanan *Backend*

REST API digunakan sebagai arsitektur layanan pada *backend* karena menyediakan mekanisme komunikasi berbasis *HTTP* antara *client* dan *server*. *Resource* sistem direpresentasikan melalui *URI/endpoint*, sedangkan operasi terhadap *resource* dilakukan menggunakan metode *HTTP* seperti *GET*, *POST*, *PUT/PATCH*, dan *DELETE*. *Response* dikembalikan dalam format yang dapat diproses *client*, seperti *JSON*, bersama *HTTP Status Code* sebagai penanda hasil pemrosesan. Penelitian empiris mengenai desain *REST* menunjukkan bahwa penggunaan aturan *REST* yang tepat dapat meningkatkan keterpahaman *Web API*[6]. Dalam penelitian ini, konsep tersebut menjadi dasar untuk merancang *endpoint* yang terpisah berdasarkan fungsi autentikasi, profil, produk, kategori, keranjang, dan pesanan. Lingkungan Implementasi *Backend*. Lingkungan implementasi *backend* terdiri atas *Node.js* dan *Express.js* sebagai lingkungan dan *framework server*, *PostgreSQL* sebagai basis data relasional, serta *Prisma ORM* sebagai penghubung antara aplikasi dan basis data. *Node.js* menjalankan kode *JavaScript* pada sisi *server*, *Express.js* mengelola *routing* dan *middleware HTTP*, *PostgreSQL* menyimpan data sistem, sedangkan *Prisma ORM* mempermudah

pengelolaan data melalui model dan skema basis data. Kombinasi tersebut digunakan agar implementasi *endpoint* dapat dilakukan secara terstruktur dan terhubung dengan basis data secara konsisten.

3. Pengujian *REST API* dengan *Postman*

Pengujian *REST API* dilakukan untuk memvalidasi perilaku *endpoint* dari sisi input dan *output* tanpa bergantung pada tampilan *frontend*. *Black Box Testing* sesuai dengan fokus penelitian karena pengujian memeriksa *request*, *response*, *Status Code*, dan keberhasilan fungsi *endpoint*. Penelitian *RESTest* menunjukkan bahwa pengujian *black-box* dapat digunakan untuk menguji *RESTful Web API* berdasarkan spesifikasi layanan[24], sedangkan *RapiTest* menunjukkan bahwa pengujian *RESTful API* dapat diarahkan pada *correctness* dan *reliability* layanan[8]. Dalam penelitian ini, *Postman* digunakan sebagai alat untuk mengirim *request* dan mencatat *Response Time*, *Status Code*, serta *success rate* pada setiap *endpoint*.

4. Lingkungan *Deployment*

Virtual Private Server (VPS) digunakan sebagai lingkungan *deployment backend* agar layanan *REST API* dapat diakses melalui jaringan. Setelah implementasi dan pengujian awal dilakukan, backend ditempatkan pada *VPS* berbasis *Ubuntu* sehingga *endpoint* dapat diuji pada lingkungan *server* yang lebih dekat dengan kondisi operasional sistem.

5. Metode *Agile Development*

Agile Development digunakan sebagai pendekatan pengembangan karena penelitian dilakukan secara bertahap dan memungkinkan perbaikan ketika ditemukan kesalahan pada proses pengujian. Tahapan yang digunakan adalah *plan* untuk menetapkan kebutuhan dan ruang lingkup, *design* untuk merancang arsitektur serta *endpoint*, *develop* untuk mengimplementasikan *backend*, *test* untuk menguji *endpoint* menggunakan *Postman*, *deploy* untuk menempatkan *backend* pada *VPS*, dan *review* untuk memvalidasi hasil. Jika hasil pengujian belum sesuai, proses kembali ke tahap perbaikan implementasi *backend* sebelum pengujian ulang. Pendekatan ini membuat prosedur penelitian dapat dipahami dan direplikasi pada penelitian selanjutnya.



Gambar 2 Metode *Agile Development*[25]

BAB III METODOLOGI

A. Waktu dan Tempat Penelitian

a. Waktu Penelitian

Waktu penelitian merupakan rentang waktu yang dialokasikan oleh peneliti untuk menyelesaikan studi secara sistematis. Tahap tersebut mencakup tahap studi literatur, pengumpulan data, perancangan dan implementasi *REST API backend*, pengujian *endpoint REST API* menggunakan *Postman*, serta penyusunan laporan akhir. Penelitian ini dijadwalkan berlangsung selama enam bulan, terhitung sejak bulan Maret 2026 hingga Agustus 2026.

b. Tempat Penelitian

Penelitian ini dilaksanakan pada UMKM Citra *Cake Sidamulih*, sebuah unit usaha mikro, kecil, dan menengah (UMKM) yang bergerak secara dinamis di sektor industri kuliner, dengan spesialisasi produksi dan penjualan berbagai varian kue. Lokasi ini dipilih karena merepresentasikan tantangan nyata yang dihadapi oleh UMKM lokal dalam menghadapi arus transformasi digital.

B. Alat dan Bahan

Alat dan bahan yang digunakan dalam penelitian ditulis menggunakan tabel seperti ini:

Tabel 2 Alat dan Bahan

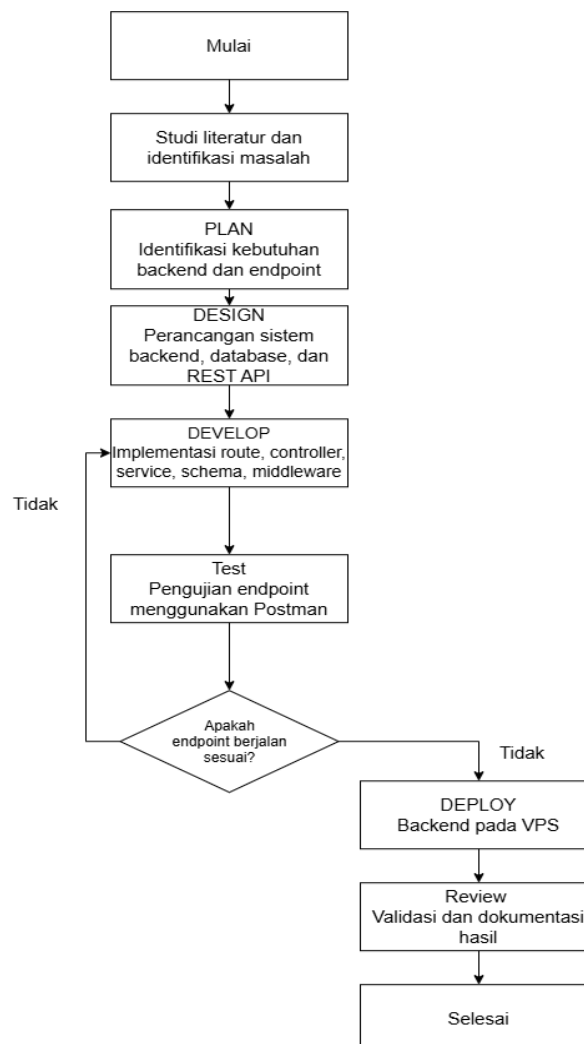
No	Alat/Bahan	Keterangan
(1)	(2)	(3)
1.	Laptop	<i>DESKTOP-I9GVOQU, Intel(R) Core(TM) i5-6300U CPU @ 2.40GHz 2.50 GHz, 8.00 GB (7.88 GB usable), A4957093-C5A5-4358-A5FE-4C41F8ECA723, 00331-10000-00001-AA869, 64-bit operating system, x64-based processor.</i>
2.	VPS	<i>Almalinux-8-9-x86_64, 1 core CPU, 512 MB RAM, 10 GB storage, unlimited.</i>
3.	<i>Visual Studio Code</i>	<i>Editor untuk perancangan dan pemeliharaan source code backend.</i>
4.	<i>Node.js & Express.js</i>	<i>Node.js sebagai runtime JavaScript dan Express.js sebagai framework untuk membangun REST API.</i>
5.	<i>PostgreSQL & Prisma ORM</i>	<i>PostgreSQL sebagai basis data relasional dan Prisma ORM sebagai penghubung aplikasi dengan basis data.</i>
6.	<i>Postman</i>	<i>Digunakan untuk pengujian endpoint REST API dengan parameter Response Time, Status Code, dan Success Rate.</i>

7.	<i>Git & GitHub</i>	Digunakan untuk pengelolaan versi dan penyimpanan <i>repository source code</i> .
8.	<i>Cloudinary</i>	Digunakan untuk penyimpanan berkas gambar yang diproses melalui <i>backend</i> .

C. Prosedur Penelitian

Prosedur penelitian disusun mengikuti tahapan *Agile Development* yang diterapkan pada implementasi *REST API backend*. Tahapan penelitian meliputi studi literatur, identifikasi kebutuhan, *plan*, *design*, *develop*, *test*, *deploy*, *review*, analisis hasil pengujian, dan penarikan kesimpulan. Jika hasil pengujian belum sesuai, proses kembali ke tahap perbaikan implementasi *backend* sebelum dilakukan pengujian ulang.

Setiap tahap yang digambarkan pada diagram alir penelitian, dijelaskan secara lebih rinci dalam sub bab penjelasan tahapan sesuai dengan diagram alir. Diagram alir ditunjukkan pada gambar berikut.



Gambar 3 *Flowchart* Penelitian

1. Studi Literatur

Tahap ini langkah awal dalam menciptakan fondasi teoritis untuk penelitian. Peneliti melakukan kajian berbagai sumber referensi seperti penelitian terdahulu tentang penerapan *REST API*, buku tentang perancangan *backend* menggunakan *Node.js*, serta dokumentasi teknis *PostgreSQL*, *Prisma ORM*, dan *Postman*. Tinjauan literatur dilakukan untuk mendapatkan pemahaman mengenai konsep *REST API*, arsitektur *backend*, manajemen basis data, metode pengujian *endpoint API*. Selain itu, studi literatur juga bertujuan untuk menyusun kerangka pemikiran yang menjadi dasar dalam implementasi dan pengujian *REST API backend* pada sistem Penjualan UMKM Citra Cake Sidamulih.

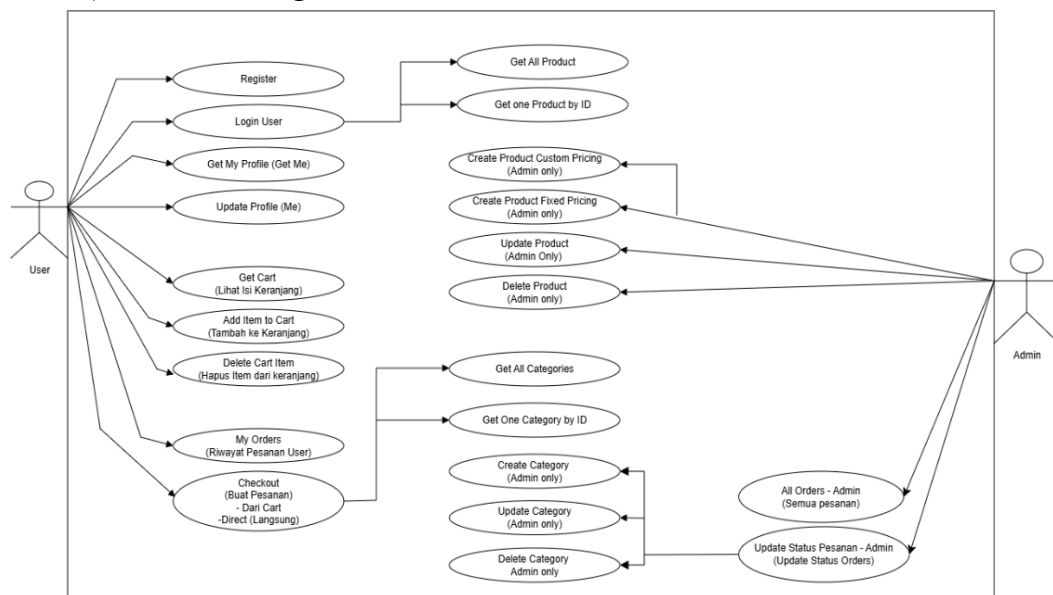
2. Identifikasi Masalah & Tujuan

Pada tahap ini, peneliti mengidentifikasi berbagai permasalahan yang muncul dalam proses penjualan UMKM Citra Cake Sidamulih, seperti pengelolaan informasi produk, kategori, pelanggan, keranjang belanja, dan pesanan yang memerlukan sistem *backend* yang terintegrasi. Berdasarkan permasalahan tersebut, tujuan penelitian dirumuskan untuk menerapkan *REST API backend* yang mampu mengelola data secara terstruktur serta melakukan pengujian *endpoint* menggunakan *Postman* untuk memastikan setiap layanan *API* dapat berfungsi dengan kebutuhan sistem melalui validasi permintaan, tanggapan, dan kode status.

3. Perancangan Sistem Web

Perancangan sistem web, dilakukan tinjauan terhadap kebutuhan sistem dan penyusunan *blueprint* yang mencakup *Use Case Diagram*, *Activity Diagram*, *Sequence Diagram*, *Class Diagram*, serta *desain endpoint REST API*. Tujuan dari tahap ini adalah untuk menyajikan gambaran tentang struktur sistem yang akan dibangun agar proses implementasi dapat dilakukan dengan terarah dan sesuai dengan kebutuhan pengguna.

1) Use Case Diagram

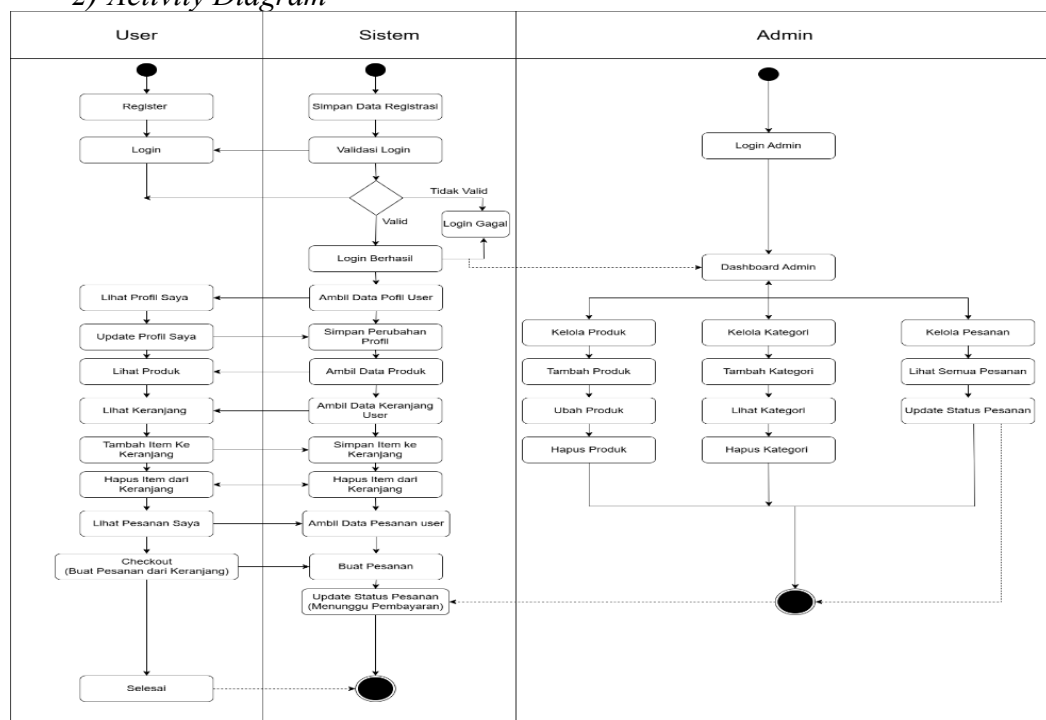


Gambar 4 Use Case Diagram

Diagram *use case* ini menggambarkan sistem *e-commerce* dengan dua aktor utama, yaitu *User* dan *Admin*. *User* sebagai individu biasa dapat menggunakan fitur autentikasi seperti pendaftaran akun, masuk, melihat *profile* pribadi (dapatkan profil saya), serta memperbarui informasi *profile*. Setelah berhasil memasuki sistem, pengguna dapat melihat daftar produk dan rincian produk berdasarkan *ID*. Selain itu, pengguna juga dapat mengatur keranjang belanja (keranjang) dengan memeriksa isi keranjang, menambahkan produk ke dalam keranjang, dan menghapus produk dari keranjang. Dalam proses transaksi, pengguna dapat mengecek riwayat pesanan (pesanan saya) dan melakukan pembayaran, baik melalui produk yang ada di keranjang maupun pembelian langsung (pembayaran langsung).

Di sisi lain, *Admin* memiliki akses yang lebih luas untuk mengatur data dalam sistem. *Admin* dapat melakukan *CRUD* (Buat, Baca, Perbarui, Hapus) pada data produk dan kategori, seperti menambahkan produk dengan harga tetap (penetapan harga tetap) atau harga khusus (harga khusus), memperbarui detail produk, serta menghapus produk yang sudah tidak digunakan lagi. Pada modul kategori, pengelola juga dapat menambah, mengedit, dan menghilangkan kategori produk. Selain itu, *Admin* bertanggung jawab untuk mengelola pesanan dengan memantau seluruh pesanan yang masuk (semua pesanan) dan memperbarui status pesanan sesuai dengan proses bisnis yang berlaku. Dengan pembagian hak akses ini, sistem dapat menjaga keamanan data dan memastikan setiap aktor hanya dapat mengakses fitur sesuai dengan perannya.

2) Activity Diagram



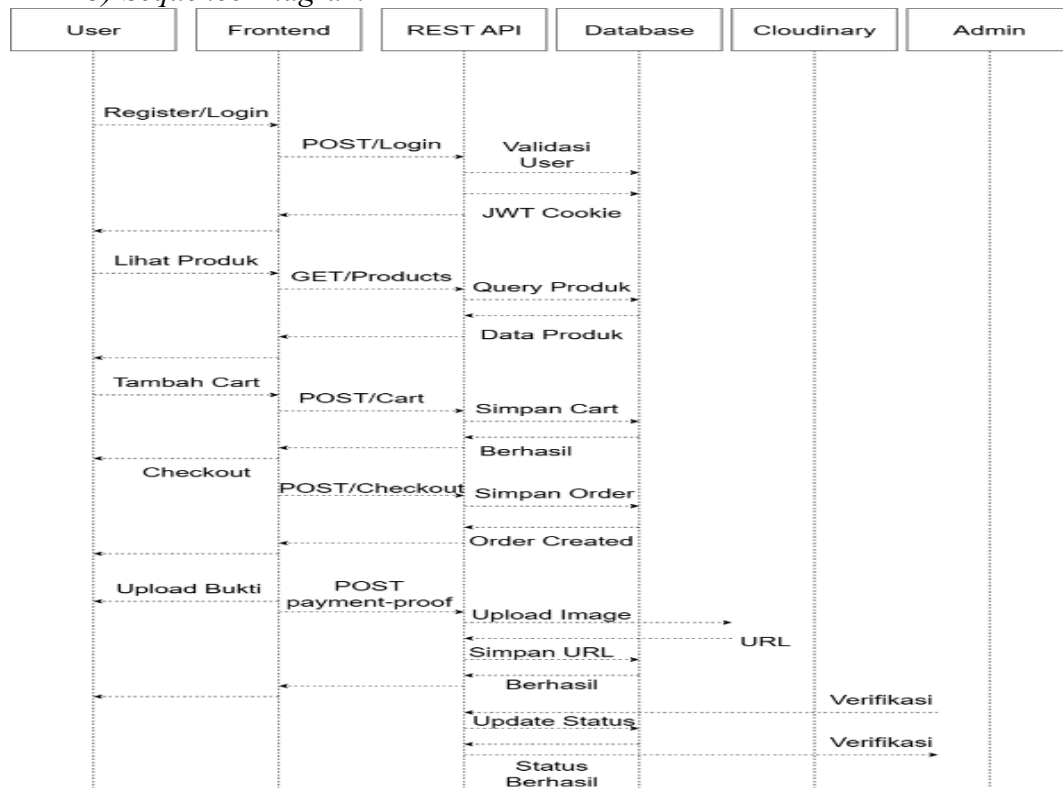
Gambar 5 Activity Diagram

Activity Diagram untuk Sistem Penjualan Citra Cake menggambarkan alur kegiatan yang dilakukan oleh *User*, Sistem, dan *Admin* dalam melaksanakan proses bisnis aplikasi. Diagram ini dimulai saat pengguna mendaftarkan akun. Data

pendaftaran yang dimasukkan oleh pengguna akan disimpan oleh sistem, dan pengguna selanjutnya dapat masuk ke dalam sistem. Sistem akan memvalidasi informasi *login* yang telah diberikan. Apabila informasi *login* tidak valid, sistem akan menampilkan pesan kegagalan login dan pengguna diminta untuk mengulangi proses masuk. Di sisi lain, jika informasi *login* valid, maka pengguna akan berhasil mengakses sistem.

Setelah berhasil masuk, pengguna bisa melihat *profile* pribadi (Dapatkan Saya) dan melakukan perubahan pada data profil (Perbarui *Profile*). Sistem akan mengambil data *profile* pengguna dari basis data dan menampilkan informasi yang diperlukan. Jika pengguna melakukan perubahan pada data profil, sistem akan menyimpan pembaruan tersebut ke dalam basis data. Selanjutnya, pengguna dapat melihat daftar produk yang tersedia. Sistem akan mengambil data produk dari basis data dan menampilkannya kepada pengguna. Pengguna pun dapat melihat isi keranjang belanja, menambah produk ke dalam keranjang, atau menghapus produk dari keranjang. Setiap aktivitas tersebut akan diproses dan disimpan oleh sistem demi memastikan bahwa data keranjang selalu *terupdate*.

3) Sequence Diagram

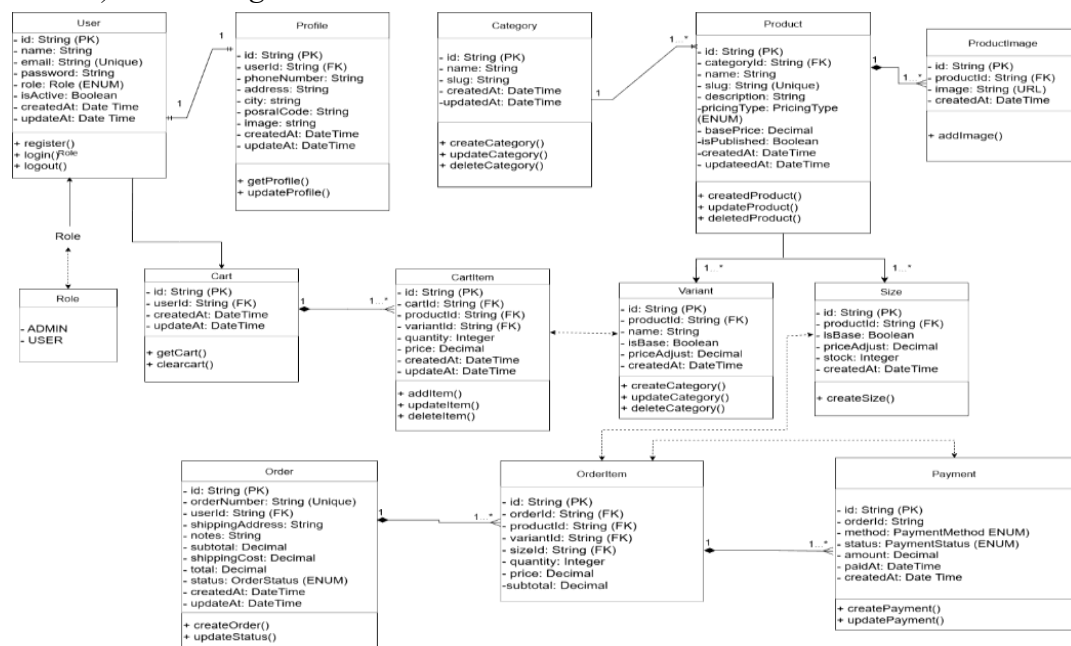


Gambar 6 Sequence Diagram

Sequence diagram ini menggambarkan alur interaksi antara pengguna, antarmuka, *REST API backend*, basis data, *cloudinary*, dan *admin* dalam sistem penjualan UMKM Citra Cake Sidamulih. Proses mulai ketika pengguna melakukan registrasi atau masuk melalui antarmuka. Selanjutnya, antarmuka mengirim *POST/login* ke *REST API* untuk memulai proses autentikasi. *Backend* kemudian memeriksa data pengguna pada basis data *PostgreSQL*. Apabila data yang dimasukkan cocok, sistem akan menghasilkan *JSON Web Token (JWT)* yang

disimpan sebagai *cookie* untuk identifikasi pengguna saat mengakses sistem. Setelah berhasil masuk, pengguna dapat melihat daftar produk melalui *endpoint GET/products*. *REST API* akan mengambil data produk melalui dan mengirimkan hasilnya kembali ke antarmuka, sehingga informasi produk dapat ditampilkan kepada pengguna. Selanjutnya, pengguna dapat menambahkan produk ke keranjang belanja melalui *endpoint POST/cart*, dan *backend* menyimpan data keranjang dalam basis data sambil mengirimkan respons bahwa proses tersebut berhasil. Ketika pengguna melakukan *checkout*, antarmuka mengirimkan permintaan *POST/checkout* ke *REST API* untuk menyimpan data pesanan ke dalam basis data hingga status pesanan berhasil dibuat (*Order Created*). Setelah menyelesaikan pembayaran, pengguna mengunggah bukti transfer melalui *endpoint POST/payment-proof*. *REST API* mengirimkan *file* gambar ke layanan *Cloudinary* untuk keperluan penyimpanan, lalu *Cloudinary* mengembalikan *URL* gambar yang kemudian disimpan dalam basis data. *Admin* kemudian melakukan verifikasi pembayaran melalui *endpoint PATCH /api/order/admin/:id/verify*, sehingga status pesanan diperbarui menjadi telah diverifikasi. Setelah itu *admin* dapat melakukan proses pengiriman melalui *endpoint PATCH/api/order/admin/:id/ship* dan mengunggah bukti pengiriman melalui *endpoint POST /api/order/admin/:id/proof-delivery*. Terakhir, pengguna melakukan konfirmasi penerimaan pesanan melalui *endpoint PATCH/api/order/admin/:id/confirm-arrival*, sehingga proses transaksi dinyatakan selesai.

4) Class Diagram



Gambar 7 Class Diagram

Class diagram tersebut menggambarkan struktur data dan hubungan antar kelas dalam sistem *e-commerce*. *Class User* menjadi entitas utama yang menyimpan informasi pengguna dan memiliki hubungan satu-ke-satu dengan *UserProfile* serta satu-ke-satu dengan *Cart*. Setiap *user* dapat melakukan banyak *Order*, sedangkan setiap *order* terdiri dari satu atau lebih *OrderItem* yang mereferensikan *Product*.

Produk dikelompokkan ke dalam *Category*, dimana satu kategori dapat memiliki banyak produk. Pada proses transaksi, setiap *order* memiliki satu *Payment* yang diproses melalui *PaymentGateway*. Selain itu, terdapat *class admin* yang berperan dalam mengelola data pengguna, produk, kategori, dan pesanan. Diagram ini juga menunjukkan atribut, metode, serta kardinalitas hubungan antar kelas, sehingga memberikan gambaran yang jelas mengenai bagaimana objek-objek dalam sistem saling berinteraksi dan mendukung proses bisnis *e-commerce* secara keseluruhan.

5) Integrasi & Deployment Sistem ke VPS

Setelah implementasi *backend* selesai, dilakukan *deployment backend* ke *Virtual Private Server (VPS)*. Tahap ini bertujuan agar *REST API* dapat diakses melalui jaringan internet dan diuji menggunakan Postman pada lingkungan *server* yang telah disiapkan. Integrasi frontend tidak menjadi fokus penelitian karena bagian *frontend* dikerjakan oleh rekan peneliti.

6) Perancangan Skenario Pengujian Menggunakan Postman

- 1) Pengujian *API* Autentikasi.
- 2) Pengujian *API Profile*.
- 3) Pengujian *API* Produk.
- 4) Pengujian *API* Kategori.
- 5) Pengujian *API* Cart.
- 6) Pengujian *API* Order.
- 7) Pengujian Respons *REST API* Menggunakan Postman.

Tabel 3 Pengujian *API* Autentikasi

No.	Fitur yang diuji	Endpoint	Method	Data Uji
1.	Registrasi Admin	<i>/api/auth/register</i>	<i>POST</i>	Name: admin, test, Email: adminTest@gmail.com , Password: admin12345, ConfirmPassword: admin123345, Role: ADMIN
2.	Registrasi User	<i>/api/auth/register</i>	<i>POST</i>	Name: user_test, Email: userTest@gmail.com , Password: user12345, confirm Password: user12345
3.	Login Admin	<i>/api/auth/login</i>	<i>POST</i>	Email: adminTest@gmail.com , Password: admin12345

4.	Login User	/api/auth/login	POST	Email: userTest@gmail.com Password: user12345
----	------------	-----------------	------	---

Tabel 4 Pengujian API Profile

No.	Fitur yang diuji	Endpoint	Method	Data diuji
1.	Get My Profile	/api/profile/me	GET	Authorization: httpOnly Cookie (user telah login)
2.	Update Profile	/api/profile/me	PUT	Name: User1_update, Phone Number: 0887772890

Tabel 5 Pengujian API Produk

No.	Fitur yang diuji	Endpoint	Method	Data diuji
1.	Get All Products	/api/products	GET	Authorization: httpOnly Cookie
2.	Get One Product by ID	/api/products/{id}	GET	ID Produk: cmoh6gusm0008e wuleqkezcfs
3.	Create Product (CUSTOM Pricing)	/api/products	POST	Data produk dengan pricingType = CUSTOM
4.	Create Product (FIXED Pricing)	/api/products	POST	Data produk dengan pricingType = FIXED.
5.	Update Product	/api/products/{id}	PUT	Perubahan data nama produk, harga, varian, dan ukuran.

6.	<i>Delete Product</i>	<i>/api/products/{id}</i>	<i>DELETE</i>	<i>ID produk yang akan di hapus.</i>
----	-----------------------	---------------------------	---------------	--------------------------------------

Tabel 6 Pengujian *API* Kategori

No.	Fitur yang diuji	Endpoint	Method	Data diuji
1.	<i>Get All Categories</i>	<i>/api/categories</i>	<i>GET</i>	<i>Authorization: http only Cookie.</i>
2.	<i>Get One Category by ID</i>	<i>/api/categories/{id}</i>	<i>GET</i>	<i>ID kategori yang valid.</i>
3.	<i>Create Category</i>	<i>/api/categories</i>	<i>POST</i>	<i>Name: Kue Lapis Legit minimum, Slug: kue-lapis-legit-minimum.</i>
4.	<i>Update Category</i>	<i>/api/categories/{id}</i>	<i>PUT</i>	<i>Name: update_categories, slug: update_categories.</i>
5.	<i>Delete Category</i>	<i>/api/categories/{id}</i>	<i>DELETE</i>	<i>ID kategori yang valid.</i>

Tabel 7 Pengujian *API* Cart

No.	Fitur yang diuji	Endpoint	Method	Data diuji
1.	<i>Get Cart</i>	<i>/api/cart</i>	<i>GET</i>	<i>Authorization: httpOnly Cookie User.</i>
2.	<i>Add Item to Cart</i>	<i>/api/cart</i>	<i>POST</i>	<i>Product ID, Variant ID, Size ID, Quantity.</i>
3.	<i>Delete Cart</i>	<i>/api/cart/{id}</i>	<i>DELETE</i>	<i>ID cart item yang valid.</i>

Tabel 8 Pengujian *API Order*

No.	Fitur yang diuji	Endpoint	Method	Data diuji
1.	<i>My Orders</i> (Riwayat Pesanan User)	<i>/api/order/my-orders</i>	<i>GET</i>	<i>Authorization: httpOnly</i> <i>Cookie User.</i>
2.	<i>All Orders</i>	<i>/api/order/admin/all</i>	<i>GET</i>	<i>Authorization: httpOnly</i> <i>Cookie admin.</i>
3.	<i>Checkout</i> (Cart)	<i>/api/order/checkout</i>	<i>POST</i>	<i>Source: cart, cartItemIds,</i> <i>shippingAddress, notes.</i>
4.	<i>Checkout</i> (Direct)	<i>/api/order/checkout</i>	<i>POST</i>	<i>Source: direct,</i> <i>productId, variantId,</i> <i>sizeId, quantity,</i> <i>shippingAddress, notes.</i>
5.	<i>Update</i> <i>Status Pesanan</i>	<i>/api/order/admin/{id}</i> <i>}/status</i>	<i>PUT</i>	<i>Status PROCESSING.</i>

Tabel 9 Pengujian Respons *REST API* Menggunakan *Postman*

No.	Fitur yang diuji	Jenis Pengujian	Parameter
1.	<i>Registrasi Admin</i>	<i>Single Track</i> (1 kali) dan <i>Multi Track</i> (5 kali).	<i>Response Time, Status</i> <i>Code, Tingkat</i> <i>keberhasilan.</i>
2.	<i>Registrasi User</i>	<i>Single Track</i> (1 kali) dan <i>Multi Track</i> (5 kali).	<i>Response Time, Status</i> <i>Code, Tingkat</i> <i>keberhasilan.</i>
3.	<i>Login Admin</i>	<i>Single Track</i> (1 kali) dan <i>Multi Track</i> (5 kali).	<i>Response Time, Status</i> <i>Code, Tingkat</i> <i>keberhasilan.</i>
4.	<i>Login User</i>	<i>Single Track</i> (1 kali) dan <i>Multi Track</i> (5 kali).	<i>Response Time, Status</i> <i>Code, Tingkat</i> <i>keberhasilan.</i>
5.	<i>Get My Profile</i>	<i>Single Track</i> (1 kali) dan <i>Multi Track</i> (5 kali).	<i>Response Time, Status</i> <i>Code, Tingkat</i> <i>keberhasilan.</i>

6.	<i>Update Profile</i>	<i>Single Track (1 kali) dan Multi Track (5 kali).</i>	<i>Response Time, Status Code, Tingkat keberhasilan.</i>
7.	<i>Get All Products</i>	<i>Single Track (1 kali) dan Multi Track (5 kali).</i>	<i>Response Time, Status Code, Tingkat keberhasilan.</i>
8.	<i>Get One Product by ID</i>	<i>Single Track (1 kali) dan Multi Track (5 kali).</i>	<i>Response Time, Status Code, Tingkat keberhasilan.</i>
9.	<i>Create Product (CUSTOM Pricing)</i>	<i>Single Track (1 kali) dan Multi Track (5 kali).</i>	<i>Response Time, Status Code, Tingkat keberhasilan.</i>
10.	<i>Create Product (FIXED Pricing)</i>	<i>Single Track (1 kali) dan Multi Track (5 kali).</i>	<i>Response Time, Status Code, Tingkat keberhasilan.</i>
11.	<i>Update Product</i>	<i>Single Track (1 kali) dan Multi Track (5 kali).</i>	<i>Response Time, Status Code, Tingkat keberhasilan.</i>
12.	<i>Delete Product</i>	<i>Single Track (1 kali) dan Multi Track (5 kali).</i>	<i>Response Time, Status Code, Tingkat keberhasilan.</i>
13.	<i>Get All Categories</i>	<i>Single Track (1 kali) dan Multi Track (5 kali).</i>	<i>Response Time, Status Code, Tingkat keberhasilan.</i>
14.	<i>Get One Category by ID</i>	<i>Single Track (1 kali) dan Multi Track (5 kali).</i>	<i>Response Time, Status Code, Tingkat keberhasilan.</i>
15.	<i>Create Category</i>	<i>Single Track (1 kali) dan Multi Track (5 kali).</i>	<i>Response Time, Status Code, Tingkat keberhasilan.</i>
16.	<i>Update Category</i>	<i>Single Track (1 kali) dan Multi Track (5 kali).</i>	<i>Response Time, Status Code, Tingkat keberhasilan.</i>
17.	<i>Delete Category</i>	<i>Single Track (1 kali) dan Multi Track (5 kali).</i>	<i>Response Time, Status Code, Tingkat keberhasilan.</i>
18.	<i>Get Cart</i>	<i>Single Track (1 kali) dan Multi Track (5 kali).</i>	<i>Response Time, Status Code, Tingkat keberhasilan.</i>

19.	<i>Add Item to Cart</i>	<i>Single Track (1 kali)</i> dan <i>Multi Track (5 kali)</i> .	<i>Response Time, Status Code, Tingkat keberhasilan.</i>
20.	<i>Delete Cart</i>	<i>Single Track (1 kali)</i> dan <i>Multi Track (5 kali)</i> .	<i>Response Time, Status Code, Tingkat keberhasilan.</i>
21.	<i>My Orders</i> (<i>Riwayat Pesanan User</i>)	<i>Single Track (1 kali)</i> dan <i>Multi Track (5 kali)</i> .	<i>Response Time, Status Code, Tingkat keberhasilan.</i>
22.	<i>All Orders</i>	<i>Single Track (1 kali)</i> dan <i>Multi Track (5 kali)</i> .	<i>Response Time, Status Code, Tingkat keberhasilan.</i>
23.	<i>Checkout (Cart)</i>	<i>Single Track (1 kali)</i> dan <i>Multi Track (5 kali)</i> .	<i>Response Time, Status Code, Tingkat keberhasilan.</i>
24.	<i>Checkout (Direct)</i>	<i>Single Track (1 kali)</i> dan <i>Multi Track (5 kali)</i> .	<i>Response Time, Status Code, Tingkat keberhasilan.</i>
25.	<i>Update Status Pesanan</i>	<i>Single Track (1 kali)</i> dan <i>Multi Track (5 kali)</i> .	<i>Response Time, Status Code, Tingkat keberhasilan.</i>

D. Analisis Data

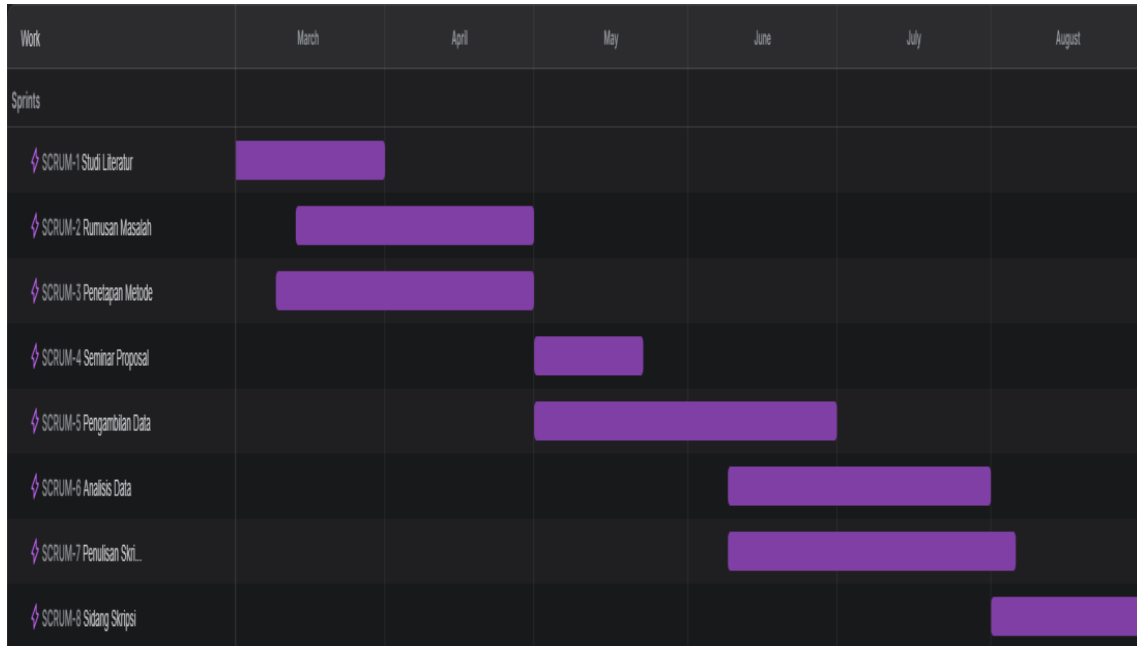
Data yang diperoleh dari hasil pengujian *REST API backend* melalui aplikasi *Postman* dianalisis secara kuantitatif. Setiap *endpoint* yang diimplementasikan dianalisis menggunakan metode *Single Track* (1 kali pengujian) dan *Multi Track* (5 kali pengujian). Parameter yang dianalisis meliputi waktu respons, *code status*, dan tingkat keberhasilan (*success rate*). Analisis waktu respons dilakukan dengan mengamati lamanya waktu yang diperlukan *server* untuk merespons setiap permintaan (*request*) yang dikirimkan melalui *Postman*. Nilai waktu respons yang didapat kemudian dibandingkan antara hasil pengujian *Single Track* dan *Multi Track* untuk menilai kestabilan performa *REST API backend*.

Analisis *code status* dilakukan untuk memastikan bahwa setiap *endpoint* mengeluarkan *code respons HTTP* yang sesuai dengan fungsi layanan yang dijalankan, seperti *code 200 OK*, *201 created*, *400 validation error*, *401 unauthorized*, *403 forbidden*, *404 not found*, *409 conflict error*, *422 invalid reference*, dan *500 internal server error*. Selanjutnya, tingkat keberhasilan (*success rate*) dihitung berdasarkan jumlah permintaan yang berhasil diproses dibandingkan dengan jumlah permintaan yang dikirimkan. Tingkat keberhasilan dihitung menggunakan rumus berikut:

$$\text{Success Rate} = \frac{\text{Jumlah Request berhasil}}{\text{Jumlah Seluruh Request}} \times 100 \%$$

Hasil analisis dari ketiga parameter tersebut kemudian disajikan dalam bentuk tabel dan dijelaskan secara deskriptif untuk mengetahui apakah implementasi *REST API backend* telah berjalan sesuai dengan kebutuhan sistem penjualan UMKM Citra Cake Sidamulih.

E. Jadwal Penelitian



Gambar 8 Jadwal Penelitian

BAB IV HASIL DAN PEMBAHASAN

A. Hasil

Implementasi REST API Backend

Dalam penelitian ini, berhasil diterapkan *REST API backend* pada Sistem Penjualan UMKM Citra Cake Sidamulih dengan menggunakan *Node.js* sebagai *runtime JavaScript*, *Express.js* sebagai kerangka kerja *backend*, *PostgreSQL* sebagai sistem pengelolaan basis data, serta *Prisma ORM* sebagai sarana komunikasi antara aplikasi dan *database*. Penerapan *backend* dilakukan untuk menyediakan layanan pertukaran data antara aplikasi *frontend* dan *database* melalui mekanisme *Representational State Transfer (REST)* sehingga setiap proses bisnis dalam sistem dapat berjalan dengan teratur.

Proses penerapan *backend* dalam pembuatan struktur *project* menggunakan *Visual Studio Code* sebagai *platform* perancangan *web*. Selanjutnya, dilakukan konfigurasi koneksi *database PostgreSQL* melalui *file .env*, kemudian model-model *database* dibuat menggunakan *Prisma Schema* sesuai kebutuhan sistem, seperti tabel Pengguna, Profil, Produk, Kategori, Keranjang, Pesanan, *OrderItem*, dan Ulasan. Setelah model selesai dirancang, dilakukan migrasi *database* menggunakan *Prisma* sehingga semua tabel berhasil dibuat pada *PostgreSQL*. Tahap selanjutnya adalah penerapan *REST API* menggunakan *Express.js*. Pada fase ini, berbagai *endpoint* diciptakan untuk mengelola proses autentikasi pengguna, pengelolaan produk, kategori, profil pengguna, keranjang belanja, transaksi pemesanan, pembayaran, hingga ulasan produk. Setiap *endpoint* dihubungkan dengan *service* dan *controller* agar pengolahan data dapat dilakukan sesuai kebutuhan sistem. Selain itu, proses autentikasi pengguna diterapkan menggunakan *JSON Web Token (JWT)* yang disimpan dalam bentuk *HTTP Only Cookie* untuk meningkatkan keamanan akses terhadap *endpoint* yang memerlukan hak akses tertentu.

Untuk mendukung proses pengunggahan gambar, seperti foto produk dan bukti pembayaran, *backend* diintegrasikan dengan layanan *Cloudinary*. File gambar yang diunggah oleh pengguna terlebih dahulu dikirim ke *Cloudinary*, kemudian sistem menyimpan *URL* hasil unggahan tersebut ke dalam *database PostgreSQL* sehingga data gambar dapat diakses kembali saat dibutuhkan. Setelah seluruh proses penerapan selesai, aplikasi *backend* di-deploy di *Virtual Private Server (VPS)* berbasis *Ubuntu* sehingga *REST API* dapat diakses melalui jaringan internet. Proses *deployment* ini memungkinkan pengujian dilakukan pada kondisi *server* yang nyata menggunakan aplikasi *Postman*. Dengan demikian, penerapan *backend* yang telah dibuat tidak hanya dapat berjalan pada lingkungan lokal (*localhost*) tetapi juga berhasil dijalankan secara daring untuk mendukung proses pengujian dan operasional sistem.

Implementasi Sistem Backend

Dalam tahap perancangan sistem penjualan UMKM Citra Cake Sidamulih, bagian *backend* dibangun menggunakan berbagai teknologi yang terintegrasi satu sama lain. Pilihan teknologi tersebut disesuaikan dengan kebutuhan sistem agar proses pengelolaan data, komunikasi lewat *REST API*, dan penerapan aplikasi dapat berjalan dengan efektif. Teknologi yang digunakan meliputi *Node.js*, *Express.js*, *PostgreSQL*, *Prisma ORM*, *JSON Web Token (JWT)*, dan *Virtual Private Server (VPS)*.

1. Node.js

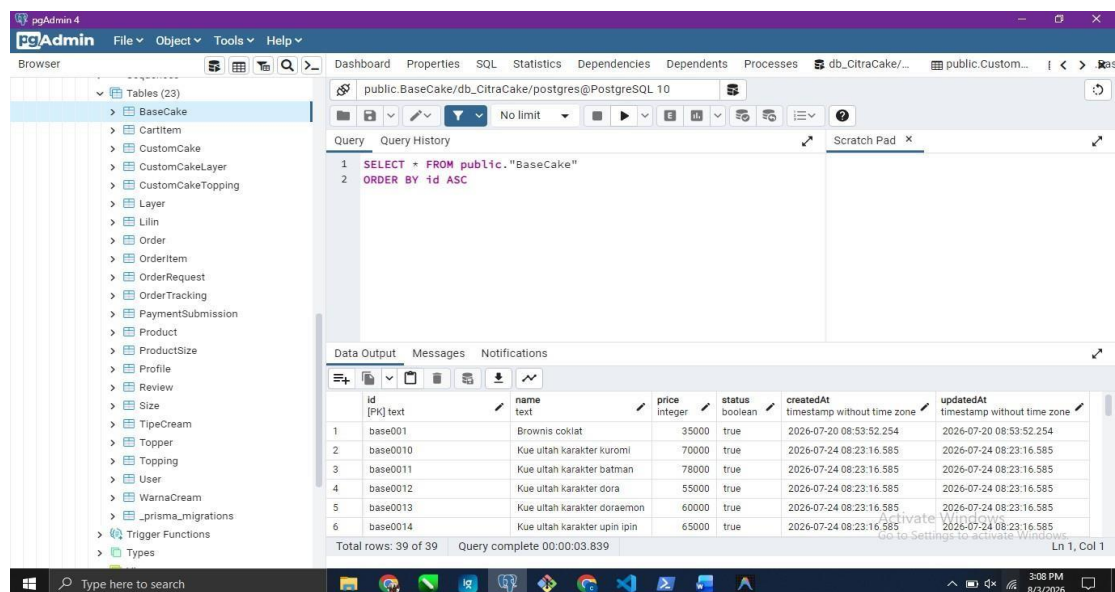
Node.js berperan sebagai lingkungan runtime yang mengoperasikan aplikasi *backend* berbasis *JavaScript* di luar *browser*. Dalam penelitian ini, *Node.js* berfungsi sebagai kerangka utama untuk menjalankan *REST API*, sehingga dapat menerima, memproses, dan memberikan balasan untuk setiap permintaan (*request*) yang diajukan oleh *frontend* maupun aplikasi *Postman*. Pemilihan *Node.js* didasarkan pada kemampuannya yang unggul dalam menangani banyak permintaan secara *asynchronous*, menjadikannya sangat cocok untuk perancangan layanan *REST API* pada sistem penjualan UMKM Citra Cake Sidamulih.

2. Express.js

Express.js adalah sebuah *framework Node.js* yang dirancang untuk menciptakan *REST API*. *Framework* ini berperan dalam mengelola *routing endpoint*, *middleware*, serta proses permintaan dan respons *HTTP*. Dalam penelitian ini, *Express.js* diimplementasikan untuk merancang *endpoint-endpoint REST API*, seperti autentikasi pengguna, pengelolaan produk, kategori, keranjang belanja, transaksi pemesanan, pembayaran, dan ulasan produk. Di samping itu, *Express.js* juga mempermudah pembagian kode berdasarkan *controller*, *service*, *middleware*, dan *route*, sehingga struktur *backend* menjadi lebih teratur.

3. PostgreSQL

PostgreSQL berperan sebagai sistem manajemen basis data relasional yang digunakan untuk menyimpan segala informasi dalam Sistem Penjualan UMKM Citra Cake Sidamulih. Informasi yang disimpan mencakup data pengguna, profil pengguna, kategori produk, produk, keranjang belanja, transaksi pemesanan, pembayaran, serta ulasan dari pelanggan. *PostgreSQL* dipilih karena kemampuannya dalam menjaga konsistensi data, mendukung transaksi *database*, serta mampu mengelola relasi antar tabel dengan efisien sehingga sesuai untuk kebutuhan sistem penjualan.



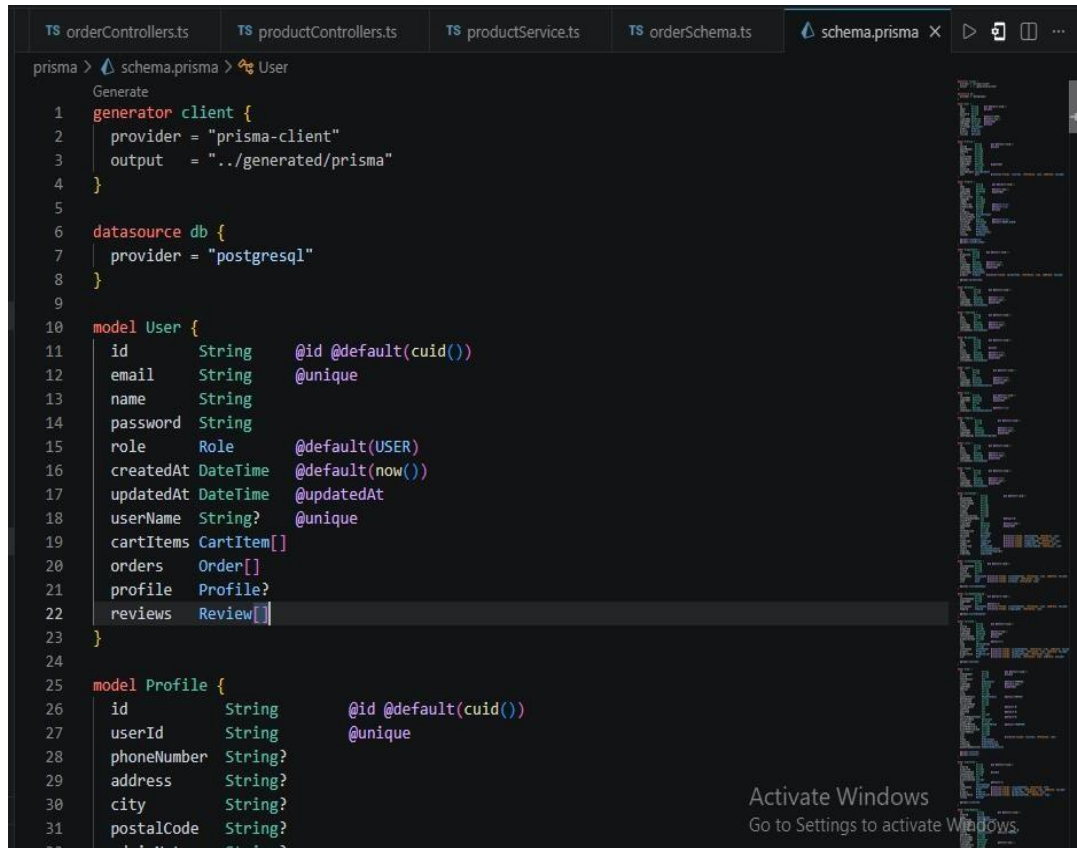
The screenshot shows the pgAdmin 4 interface. On the left, a tree view lists database objects including Tables (23), BaseCake, CartItem, CustomCake, CustomCakeLayer, Layer, Liliin, Order, OrderItem, OrderRequest, OrderTracking, PaymentSubmission, Product, ProductSize, Profile, Review, Size, TipeCream, Topper, Topping, User, WarnaCream, _prisma_migrations, Trigger Functions, and Types. The main pane displays a SQL query: `SELECT * FROM public."BaseCake" ORDER BY id ASC`. Below the query, the 'Data Output' tab shows the results of the query in a table format. The table has 6 columns: id, name, price, status, createdAt, and updatedAt. The results show 6 rows of data for the 'BaseCake' table.

id	name	price	status	createdAt	updatedAt
base001	Brownis coklat	35000	true	2026-07-20 08:53:52.254	2026-07-20 08:53:52.254
base0010	Kue ultah karakter kuromi	70000	true	2026-07-24 08:23:16.585	2026-07-24 08:23:16.585
base0011	Kue ultah karakter batman	78000	true	2026-07-24 08:23:16.585	2026-07-24 08:23:16.585
base0012	Kue ultah karakter dora	55000	true	2026-07-24 08:23:16.585	2026-07-24 08:23:16.585
base0013	Kue ultah karakter doraemon	60000	true	2026-07-24 08:23:16.585	2026-07-24 08:23:16.585
base0014	Kue ultah karakter upin ipin	65000	true	2026-07-24 08:23:16.585	2026-07-24 08:23:16.585

Gambar 9 Database

4. Prisma ORM

Prisma ORM berfungsi sebagai jembatan penghubung antara aplikasi *backend* dan basis data *PostgreSQL*. Dengan *Prisma ORM*, pengelolaan data menjadi lebih mudah karena para perancang tidak perlu menulis *query SQL* secara manual. Dalam studi ini, seluruh model basis data direncanakan menggunakan *schema prisma*, lalu dilakukan migrasi sehingga tabel-tabel dalam basis data terbentuk secara otomatis. Selanjutnya, *Prisma Client* dimanfaatkan untuk melakukan operasi *Create, Read, Update, dan Delete (CRUD)* terhadap informasi yang diperlukan oleh *REST API*.



Gambar 10 *Schema Prisma*

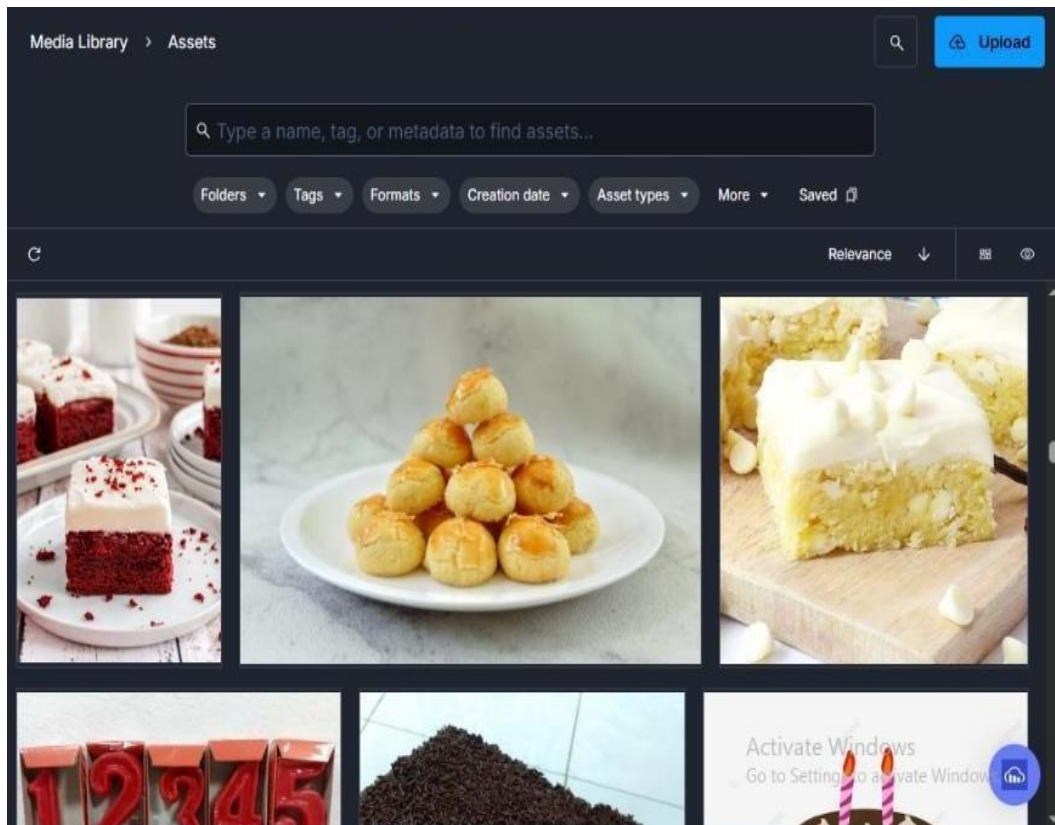
5. JSON Web Token (JWT)

JSON Web Token (JWT) berfungsi sebagai sarana autentikasi pengguna dalam sistem. Setelah *login* berhasil, *backend* menciptakan *token* autentikasi yang disimpan sebagai *HTTP Only Cookie*. *Token* ini dimanfaatkan untuk mengonfirmasi identitas pengguna saat mengakses *endpoint* yang memerlukan izin khusus. Dengan mengaplikasikan *JWT*, sistem mampu membedakan hak akses antara *Admin, Owner*, dan Pengguna, sehingga keamanan data dan operasi bisnis dalam sistem dapat terjamin.

6. Cloudinary

Dalam penelitian ini, *Cloudinary* dimanfaatkan untuk menyimpan berkas gambar seperti foto produk, bukti pembayaran, bukti pengiriman, dan juga foto ulasan pelanggan. Ketika pengguna *meng-upload* gambar melalui *endpoint REST API*, *backend* akan mengirimkan berkas tersebut ke *Cloudinary*. Setelah proses unggah

selesai, *Cloudinary* akan mengembalikan *URL* gambar yang kemudian disimpan dalam *database PostgreSQL*. Dengan cara ini, penyimpanan berkas menjadi lebih efisien karena gambar tidak disimpan langsung di *server* aplikasi.



Gambar 11 *Cloudinary*

7. Virtual Private Server

Virtual Private Server (VPS) difungsikan sebagai sarana penerapan *backend* agar *REST API* bisa diakses melalui internet. Dalam studi ini, *backend* yang sebelumnya beroperasi di lingkungan *localhost* telah berhasil dipindahkan ke *VPS* yang menggunakan sistem operasi *Ubuntu*, sehingga semua endpoint *REST API* dapat diakses menggunakan alamat *IP server*. Penerapan di *VPS* memungkinkan pengujian dilakukan dengan *Postman* dalam kondisi *server* yang nyata. Dengan cara ini, hasil pengujian yang didapatkan dapat mencerminkan kinerja *REST API backend* saat dijalankan di lingkungan produksi.

Implementasi *Endpoint REST API*

Tabel 10 Implementasi *Endpoint REST API*

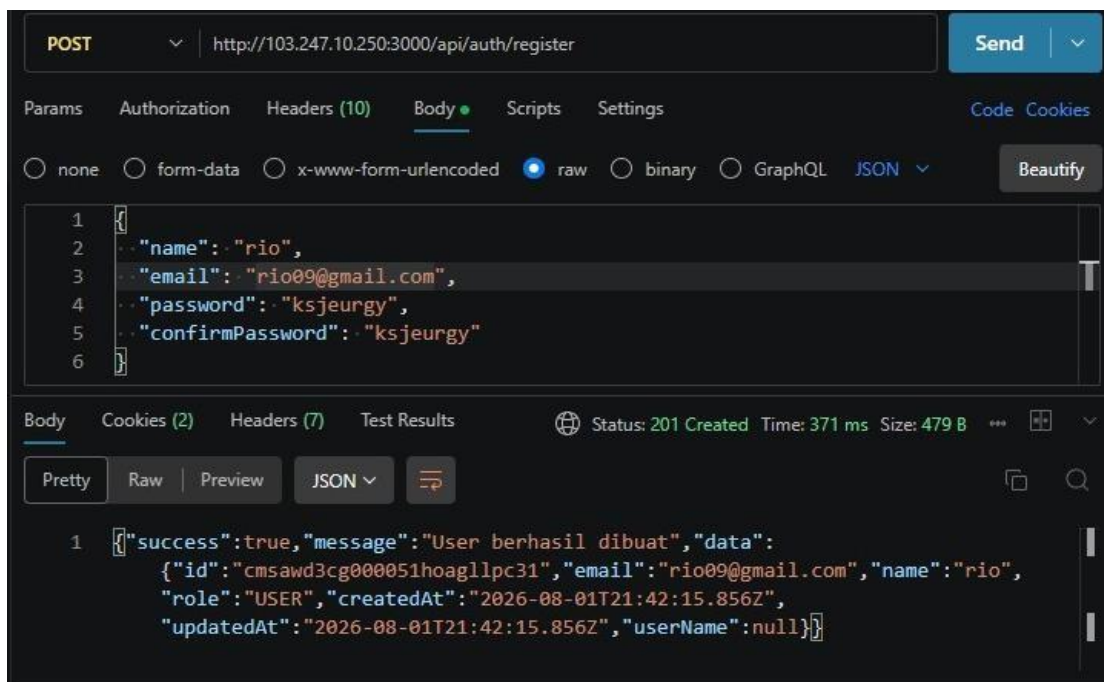
No.	Endpoint	Method	Fungsi
1.	<i>POST/api/auth/register</i>	<i>POST</i>	Registrasi <i>User</i> .
2.	<i>POST/api/auth/login</i>	<i>POST</i>	<i>Login User</i> dan <i>Admin</i> .
3.	<i>GET/api/products</i>	<i>GET</i>	Menampilkan seluruh daftar produk yang tersedia.
4.	<i>GET/api/products/:id</i>	<i>GET</i>	Digunakan untuk menampilkan

			informasi detail suatu produk berdasarkan <i>ID</i> produk yang dipilih.
5.	<i>POST/api/cart</i>	<i>POST</i>	Digunakan untuk menambahkan kue kustom ke dalam keranjang pengguna.
6.	<i>GET/api/cart</i>	<i>GET</i>	Digunakan untuk menampilkan daftar produk yang telah ditambahkan ke dalam keranjang pengguna.
7.	<i>POST/api/order/checkout</i>	<i>POST</i>	Digunakan untuk membuat pesanan berdasarkan produk yang terdapat dikeranjang belanja beserta data pengiriman dan metode pembayaran.
8.	<i>POST/api/order/my-orders/:id/payment-proof</i>	<i>POST</i>	Digunakan untuk mengunggah bukti pembayaran melalui <i>transfer bank</i> sebagai proses verifikasi pembayaran pesanan.
9.	<i>GET/api/order/my-orders</i>	<i>GET</i>	Digunakan oleh admin untuk melihat seluruh data pesanan yang masuk dari pengguna.
10.	<i>PATCH/api/order/admin/:id/verify</i>	<i>PATCH</i>	Digunakan oleh <i>admin</i> untuk memverifikasi pembayaran dan mengubah status pesanan menjadi telah diverifikasi atau ditolak.
11.	<i>PATCH/api/order/admin/:id/ship</i>	<i>POST</i>	Digunakan oleh <i>admin</i> untuk memperbarui status pesanan menjadi sedang dikirim.
12.	<i>POST/api/order/admin/:id/proof-delivery</i>	<i>POST</i>	Digunakan oleh admin untuk mengunggah bukti bahwa pesanan telah dikirim kepada pelanggan.
13.	<i>PATCH/api/order/admin/:id/confirm-arrival</i>	<i>POST</i>	Digunakan oleh admin untuk mengubah status pesanan

			menjadi selesai (<i>Completed</i>) setelah pesanan diterima pelanggan.
14.	<i>POST/api/reviews</i>	<i>POST</i>	Digunakan oleh pengguna untuk memberikan penilaian dan ulasan terhadap produk yang telah selesai dipesan.

a. Implementasi Modul Autentikasi

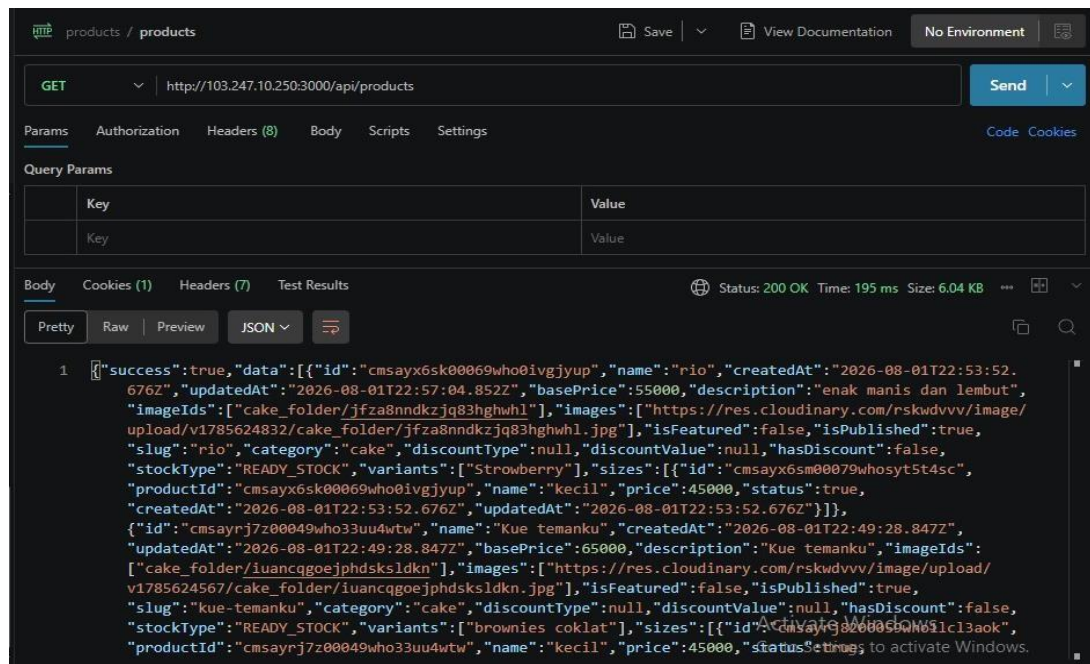
Modul autentikasi mencakup *endpoint POST /api/auth/register* dan *POST /api/auth/login*. *Endpoint* pendaftaran berfungsi untuk menyimpan informasi pengguna baru ke dalam *database PostgreSQL*, sedangkan *endpoint log* masuk bertujuan untuk melakukan proses verifikasi identitas pengguna. Setelah proses *log in* sukses, *backend* menghasilkan *JSON Web Token (JWT)* yang disimpan sebagai *HTTP Only Cookie* agar pengguna dapat mengakses *endpoint* yang memerlukan autentikasi.



Gambar 12 Register

b. Implementasi Modul Produk

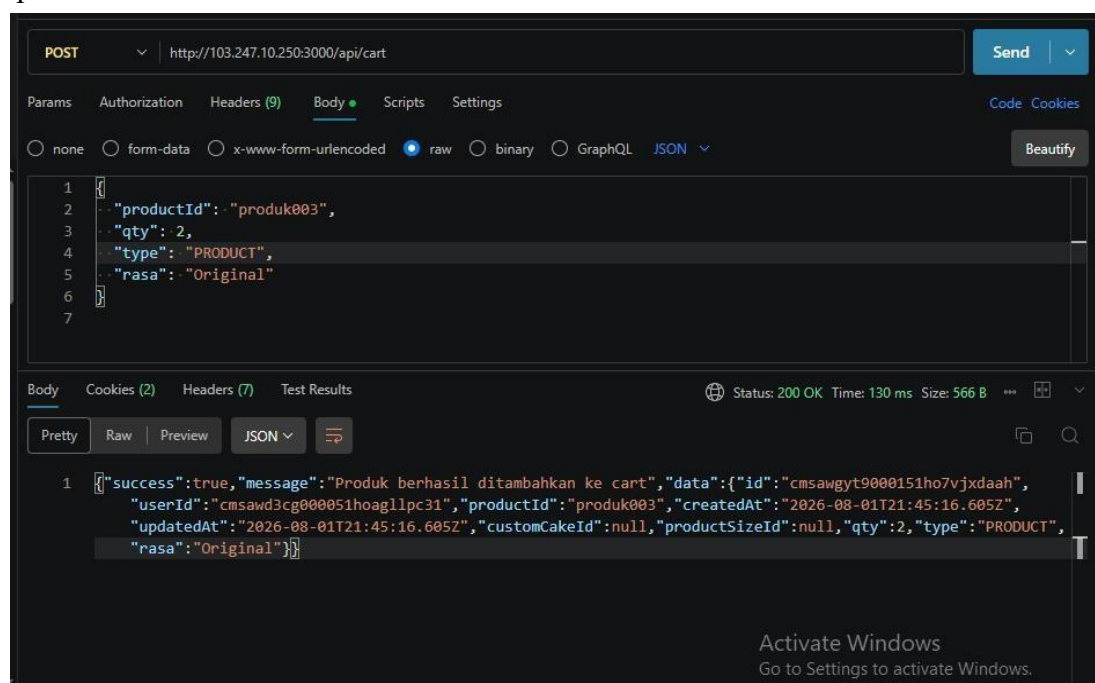
Komponen produk mencakup *endpoint GET/api/products* dan *GET /api/products/:id*. *Endpoint* ini berfungsi untuk menampilkan katalog produk serta rincian spesifik produk berdasarkan *ID*. Informasi produk diambil dari basis data *PostgreSQL* dengan bantuan *Prisma ORM* dan kemudian disampaikan dalam format *JSON* kepada *client*.



Gambar 13 *Products*

c. Implementasi Modul Keranjang

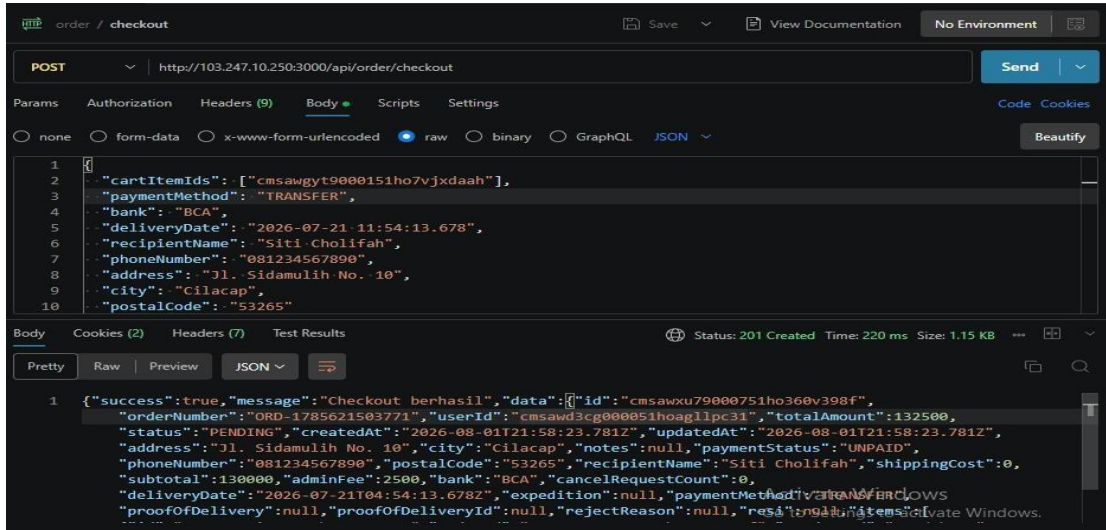
Modul keranjang belanja terdiri dari *endpoint* `POST/api/cart` dan `GET/api/cart`. *Endpoint* ini berfungsi untuk menambahkan *item* ke dalam keranjang serta menampilkan daftar *item* yang telah dipilih oleh pengguna sebelum melanjutkan ke proses *checkout*.



Gambar 14 *Cart*

d. Implementasi Modul Pemesanan

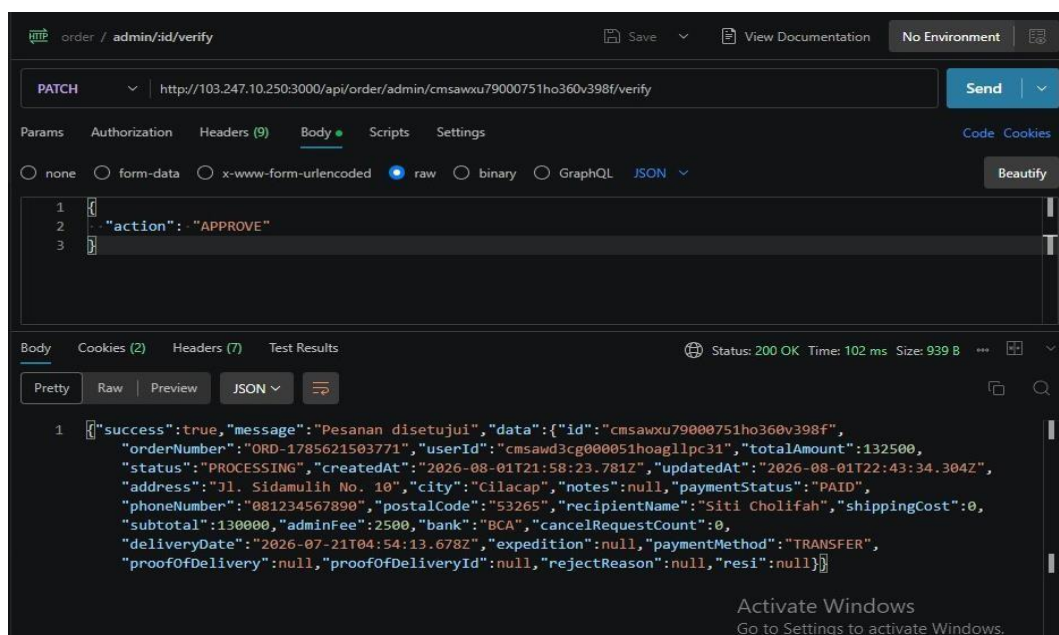
Modul pemesanan mencakup *endpoint* *POST/api/order/checkout*, *GET/api/order/my-orders*, dan *POST/api/order/my-orders/:id/payment-proof*. Modul ini berfungsi untuk melakukan transaksi pemesanan, menampilkan sejarah pemesanan pengguna, serta *upload* bukti pembayaran melalui layanan *Cloudinary*.



Gambar 15 Checkout

e. Implementasi Modul Administrasi Pesanan

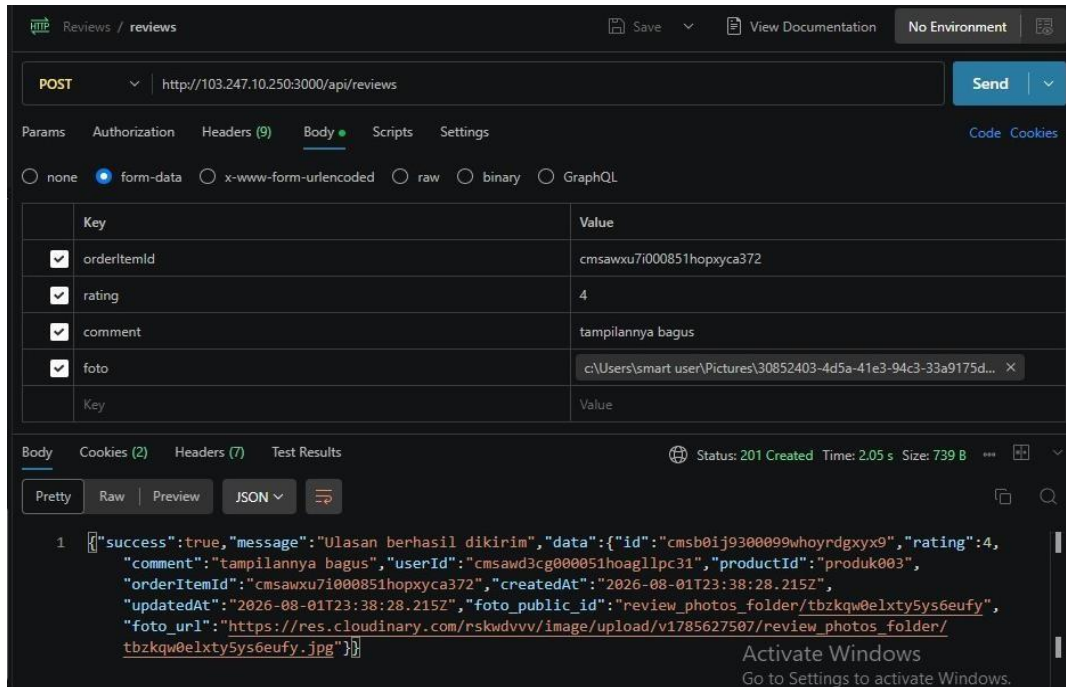
Modul administrasi pesanan mencakup *endpoint* *PATCH/verify*, *PATCH/ship*, *POST/proof-delivery*, dan *PATCH/confirm-arrival*. *Endpoint* ini dimanfaatkan oleh admin untuk memvalidasi pembayaran, memperbarui status pengiriman, mengunggah bukti pengiriman, serta memastikan bahwa pesanan telah selesai.



Gambar 16 Pesanan

f. Implementasi Modul Ulasan Produk

Modul ulasan mencakup *endpoint POST/api/reviews* yang memungkinkan pengguna untuk memberikan penilaian dan kritik terhadap produk yang telah diterima saat status pesanan berubah menjadi selesai.



Gambar 17 Review

Hasil Pengujian REST API Menggunakan Postman

Setelah proses implementasi *REST API backend* selesai dilaksanakan, tahap selanjutnya adalah melakukan pengujian terhadap semua *endpoint* menggunakan aplikasi *Postman*. Pengujian ini bertujuan untuk memastikan bahwa setiap *endpoint REST API* yang telah diterapkan dapat berfungsi sesuai dengan kebutuhan sistem penjualan UMKM Citra Cake Sidamulih. Selain menguji fungsi *endpoint*, pengujian juga bertujuan untuk mengevaluasi performa layanan backend berdasarkan parameter waktu respons, *code status*, dan tingkat keberhasilan (*success rate*). Pengujian dilakukan terhadap *backend* yang telah berhasil *dideploy* pada *Virtual Private Server (VPS)* sehingga semua permintaan dikirimkan melalui alamat *server* yang dapat diakses secara daring. Dengan cara ini, proses pengujian tidak dilakukan di lingkungan *localhost*, melainkan pada kondisi *server* yang sebenarnya sehingga hasil pengujian dapat merefleksikan performa *REST API backend* ketika digunakan oleh pengguna. Semua permintaan dikirimkan menggunakan metode *HTTP* sesuai dengan fungsi masing-masing *endpoint*, yaitu *GET*, *POT*, dan *PATCH*.

Dalam penelitian ini, tersedia dua skenario pengujian, yakni *Single Track* dan *Multi Track*. Pengujian *Single Track* dapat dilakukan dalam mengirimkan satu permintaan pada setiap *endpoint* memastikan bahwa *endpoint* berfungsi memberikan respons yang sesuai fungsi diharapkan. Untuk selanjutnya, dilakukan pengujian *Multi Track* dengan mengirimkan lima permintaan secara berturut-turut pada setiap *endpoint* untuk mengevaluasi kestabilan layanan *backend* saat menerima beberapa permintaan

secara berulang. Parameter yang diamati dalam kedua skenario tersebut mencakup waktu *respons*, *code status*, dan tingkat keberhasilan. Hasil pengujian diperoleh kemudian dianalisis untuk mengetahui tingkat keberhasilan penerapan *REST API backend*. Nilai waktu *respons* digunakan untuk mengetahui kecepatan *backend* dalam memberikan *respons* terhadap permintaan yang dikirimkan, *code status* digunakan untuk memastikan bahwa setiap *endpoint* menghasilkan *HTTP* yang sesuai dengan standar protokol *HTTP*, sedangkan tingkat keberhasilan digunakan untuk mengetahui *persentase* keberhasilan *endpoint* dalam memproses permintaan selama proses pengujian berlangsung.

Pengujian Single Track

Pengujian *Single Track* adalah untuk memastikan bahwa setiap *endpoint* yang telah diimplementasikan berfungsi sesuai dengan tugasnya dan memberikan *respons* yang tepat ketika menerima satu permintaan dari *client*. Semua *endpoint* diuji secara berurutan mengikuti alur penggunaan sistem mulai dari proses autentikasi pengguna, manajemen produk, keranjang belanja, transaksi pemesanan, pembayaran, hingga proses administrasi oleh *admin* dan pemberi ulasan produk. Pada setiap *endpoint*, terdapat tiga *parameter* yang diamati dalam pengujian, yaitu waktu *respons*, *status code*, dan tingkat keberhasilan. Waktu *respons* digunakan untuk mengukur durasi yang diperlukan *backend* dalam proses permintaan hingga mengirimkan *respons* kepada *client*. *Code status* digunakan untuk memastikan bahwa setiap *endpoint* mengeluarkan *code HTTP* yang sesuai dengan standar, sementara tingkat keberhasilan digunakan untuk mengetahui apakah permintaan berhasil diproses tanpa mengalami kegagalan.

Langkah-langkah Pengujian *Single Track* :

1. Pengujian *Single Track* dilakukan dengan langkah-langkah sebagai berikut:
2. Menjalankan *backend* yang telah dideploy pada *VPS* sehingga seluruh *endpoint REST API* dapat diakses melalui jaringan internet.
3. Membuka aplikasi *Postman* kemudian mengimpor *collection endpoint REST API* sistem penjualan UMKM Citra Cake Sidamulih.
4. Melakukan pengujian terhadap setiap *endpoint REST API* secara berurutan dengan mengirimkan satu kali *request* pada masing-masing *endpoint*.
5. Mengamati hasil *reponse* yang diberikan oleh *backend*, meliputi:
 - *Response Time*
 - *Status code*
 - *Response Body*
6. Mencatat hasil pengujian ke dalam tabel untuk dianalisis.

Tabel 11 Pengujian *Single Track*

No.	Endpoint	Reponse Time	Status Code	Success Rate
1.	POST/api/auth/register	371 ms	201 Created	100%
2.	POST/api/auth/login	225 ms	200 OK	100%
3.	GET/api/auth/me	146 ms	200 OK	100%
4.	GET/api/auth/products	89 ms	200 OK	100%

5.	<i>GET/api/products/:id</i>	94 ms	200 OK	100%
6.	<i>POST/api/cart</i>	130 ms	200 OK	100%
7.	<i>GET/api/cart</i>	75 ms	200 OK	100%
8.	<i>POST/api/order/checkout</i>	220 ms	201 Created	100%
9.	<i>POST/api/order/my-orders/:id/payment-proof</i>	3.65 s	201 Created	100%
10.	<i>GET/api/order/my-orders</i>	226 ms	200 OK	100%
11.	<i>GET/api/order/admin/all</i>	270 ms	200 OK	100%
12.	<i>PATCH/api/order/admin/:id/verify</i>	102 ms	200 OK	100%
13.	<i>PATCH/api/order/admin/:id/ship</i>	78 ms	200 OK	100%
14.	<i>POST/api/order/admin/:id/proof-delivery</i>	3.02 s	200 OK	100%
15.	<i>PATCH/api/order/admin/:id/confirm-arrival</i>	79 ms	200 OK	100%
16.	<i>POST/api/reviews</i>	2.05 s	201 OK	100%

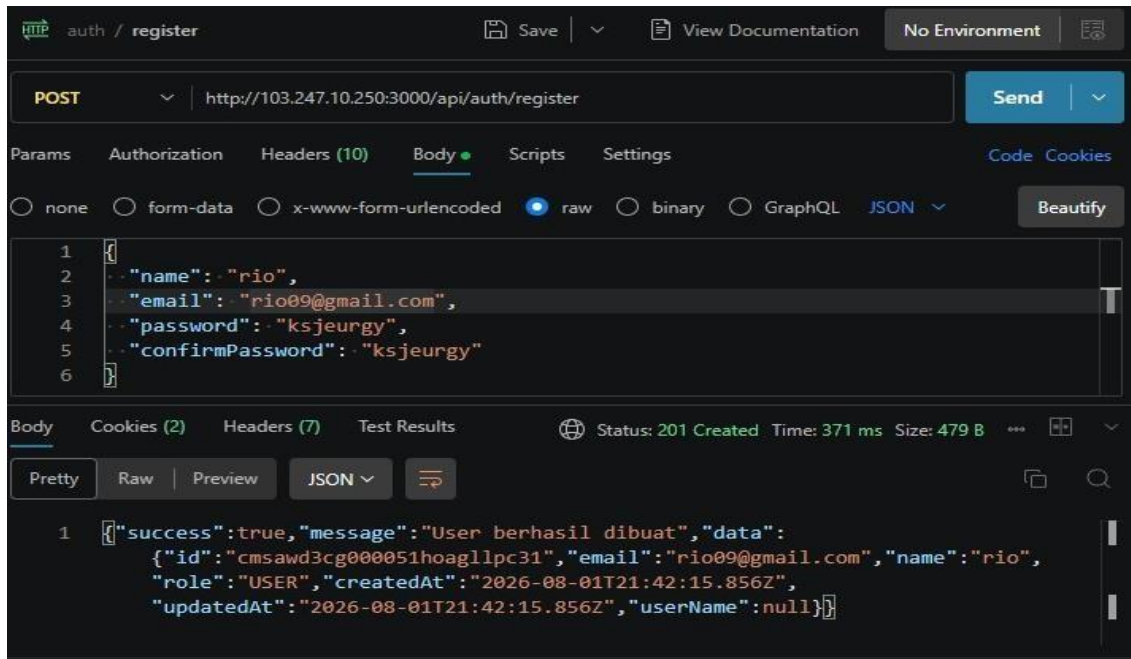
Berdasarkan hasil pengujian *Single Tracke* pada Tabel 11, seluruh *endpoint REST API backend* berhasil dijalankan menggunakan aplikasi *Postman*. Pengujian dilakukan dengan mengirimkan satu kali *request* pada setiap *endpoint* untuk mengetahui *Response Time*, *Status Code*, dan *Success Rate*. Hasil pengujian menunjukkan bahwa seluruh *endpoint* memperoleh *Success Rate* sebesar 100%, sehingga dapat disimpulkan bahwa seluruh fungsi *REST API* yang diimplementasikan pada Sistem Penjualan UMKM Citra Cake Sidamulih berjalan sesuai dengan kebutuhan fungsional sistem.

Berdasarkan nilai *Response Time*, sebagian besar *endpoint* mampu memberikan respons dengan waktu kurang dari 500 ms. *Endpoint GET/api/cart* memperoleh waktu respons tercepat sebesar 75 ms, sedangkan *endpoint GET/api/products*, *GET/api/products/:id*, dan *PATCH/api/order/admin/:id/ship* juga menunjukkan waktu respons yang relatif cepat, yaitu masing-masing sebesar 89 ms, 94 ms, dan 78 ms. Hasil tersebut menunjukkan bahwa proses autentikasi, pengambilan data, serta perubahan status pesanan dapat diproses secara efisien oleh *backend*.

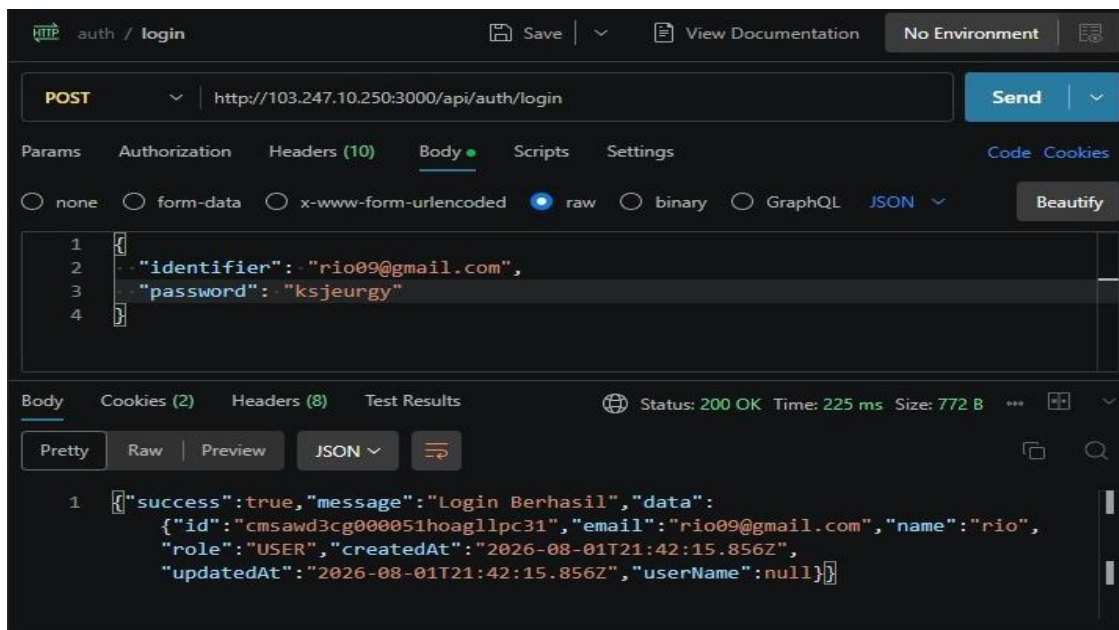
Beberapa *endpoint* memiliki waktu respons yang relatif lebih lama dibandingkan *endpoint* lainnya, yaitu *POST/api/order/my-orders/:id/payment-proof* sebesar 3.65 s, *POST/api/order/admin/:id/proof-delivery* sebesar 3.02 s, dan *POST/api/reviews* sebesar 2.05 s. Waktu respons yang lebih tinggi tersebut disebabkan karena *endpoint* tersebut melakukan proses unggah *file* gambar ke layanan *Cloudinary* serta penyimpanan data transaksi ke dalam basis data *PostgreSQL*, sehingga membutuhkan waktu pemrosesan yang lebih besar dibandingkan *endpoint* yang hanya melakukan proses pembacaan atau pembaruan data.

Selain itu, seluruh *endpoint* menghasilkan *Status Code* yang sesuai dengan standar *REST API*. *Endpoint* yang melakukan proses penambahan data, seperti *POST/api/auth/register*, *POST/api/order/checkout*, *POST/api/order/myorders/:id/p*

ayment-proof, dan *POST/api/review* menghasilkan *Status Code* 201 Created, sedangkan endpoint lainnya menghasilkan *Status Code* 200 OK yang menunjukkan bahwa permintaan berhasil diproses oleh server. Berdasarkan hasil tersebut dapat disimpulkan bahwa implementasi *REST API backend* pada Sistem Penjualan UMKM Citra Cake Sidamulih telah berjalan dengan baik dan mampu memberikan respons yang sesuai terhadap setiap permintaan yang dikirimkan melalui *Postman*.



Gambar 18 Tampilan *Register*



Gambar 19 Tampilan *Login*

Pengujian Multi Track

Pengujian *Multi Track* dilakukan untuk mengetahui konsistensi performa *REST API backend* ketika menerima beberapa permintaan (*request*) secara berulang. Pada penelitian ini, setiap *endpoint REST API* diuji sebanyak lima kali menggunakan aplikasi *Postman*. Parameter yang diamati meliputi *Response Time*, *Status Code*, dan *Success Rate*. Pengujian dilakukan terhadap *backend* yang telah dideploy pada *Virtual Private Server (VPS)* sehingga seluruh *endpoint* dapat diakses melalui jaringan internet. Pada *endpoint* yang hanya melakukan proses pembacaan data (*read operation*), seperti *Login*, *Get Profile*, *Get Products*, *Get Product by ID*, *Get Cart*, dan *Get My Orders*, pengujian dilakukan dengan mengirimkan lima kali request menggunakan data yang sama. Sementara itu, pada *endpoint* yang melakukan proses penambahan maupun perubahan data (*create dan update operation*), seperti *Register*, *Checkout*, *Upload Bukti Pembayaran*, *Verifikasi Pesanan*, *Pengiriman Pesanan*, *Upload Bukti Pengiriman*, *Konfirmasi Pesanan*, dan *Review Produk*, digunakan data yang berbeda pada setiap percobaan agar setiap request tetap valid sesuai dengan aturan bisnis sistem.

Langkah-langkah Pengujian *Multi Track*:

1. Menjalankan *backend* yang telah berhasil dideploy pada *VPS*.
2. Membuka aplikasi *Postman*.
3. Memilih *endpoint REST API* yang akan diuji.
4. Mengirimkan lima kali request pada *endpoint* yang sama.
5. Untuk *endpoint* yang memerlukan data unik, digunakan data yang berbeda pada setiap pengujian.
6. Mencatat hasil setiap pengujian berupa:
 - *Response Time*
 - *Status Code*
 - *Success Rate*
7. Menghitung rata-rata *Response Time* dari lima kali pengujian.
8. Menganalisis hasil pengujian untuk mengetahui konsistensi performa *backend*.

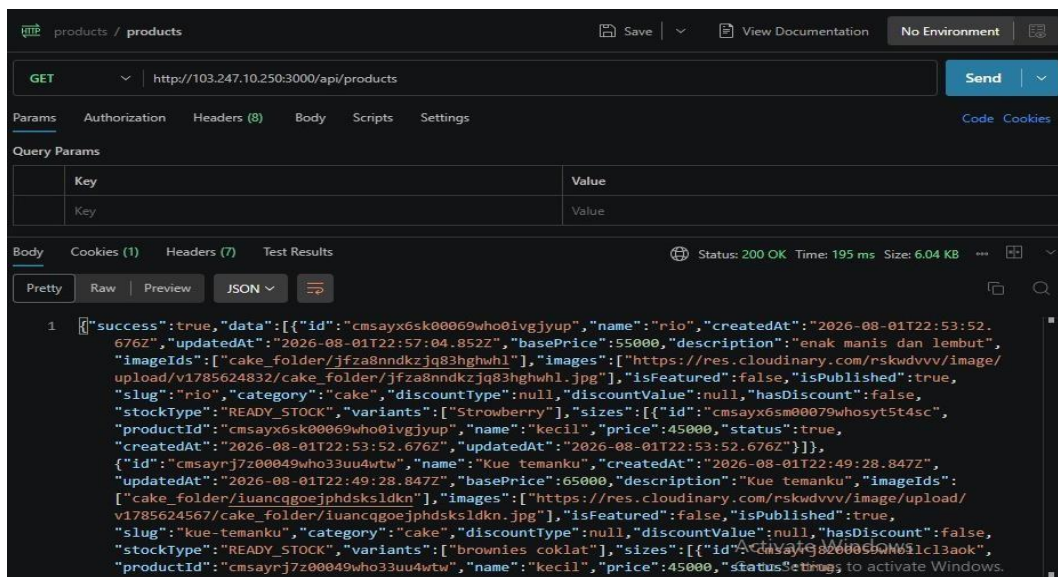
Tabel 12 Pengujian Multi Track

No.	Endpoint	Rata-rata Response Time	Status Code	Success Rate
1.	POST/api/auth/register	1.209 ms	201 Created	100%
2.	POST/api/auth/login	978.6 ms	200 OK	100%
3.	GET/api/auth/me	315.2 ms	200 OK	100%
4.	GET/api/auth/products	474 ms	200 OK	100%

5.	<i>GET/api/products/:id</i>	<i>348 ms</i>	<i>200 OK</i>	<i>100%</i>
6.	<i>GET/api/cart</i>	<i>595.2 ms</i>	<i>200 OK</i>	<i>100%</i>
7.	<i>GET/api/order/my-orders</i>	<i>671.4 ms</i>	<i>200 OK</i>	<i>100%</i>
8.	<i>GET/api/products/category/:category</i>	<i>452.2 ms</i>	<i>200 OK</i>	<i>100%</i>

Berdasarkan hasil pengujian *Multi Track* pada Tabel 12, pengujian dilakukan dengan mengirimkan lima kali *request* pada setiap *endpoint* menggunakan aplikasi *Postman* untuk mengetahui konsistensi performa *REST API backend*. Parameter yang diamati meliputi *Response Time*, *Status Code*, dan *Success Rate*. Hasil pengujian menunjukkan bahwa seluruh *endpoint* memperoleh *Success Rate* sebesar 100%, yang menandakan bahwa seluruh *request* berhasil diproses oleh *backend* tanpa mengalami kegagalan selama lima kali pengujian. Selain itu, seluruh *endpoint* menghasilkan *Status Code* yang sesuai dengan standar *REST API*, yaitu 200 OK untuk proses pengambilan data (*GET*) dan 201 *Created* untuk proses registrasi pengguna (*POST*), sehingga menunjukkan bahwa implementasi *REST API* telah berjalan sesuai dengan fungsi yang dirancang.

Berdasarkan nilai rata-rata *Response Time*, *endpoint GET/api/auth/me* memperoleh waktu respons tercepat yaitu 315.2 ms, diikuti oleh *GET/api/products/:id* sebesar 348 ms dan *GET/api/products/category/:category* sebesar 452,2 ms. *Endpoint GET /api/products* memiliki rata-rata waktu respons sebesar 474 ms, sedangkan *GET /api/cart* dan *GET /api/order/my-orders* masing-masing memperoleh 595,2 ms dan 671,4 ms. *Endpoint* yang memerlukan proses autentikasi, yaitu *POST /api/auth/login*, memiliki rata-rata waktu respons sebesar 978,6 ms, sedangkan *POST /api/auth/register* menghasilkan waktu respons paling tinggi yaitu 1.209 ms. Waktu respons yang relatif lebih besar pada proses registrasi dan *login* dipengaruhi oleh proses validasi data pengguna, autentikasi menggunakan *JSON Web Token (JWT)*, serta penyimpanan data pengguna ke dalam basis data *PostgreSQL* melalui *Prisma ORM*. Meskipun demikian, seluruh *endpoint* masih dapat diproses dengan tingkat keberhasilan 100%, sehingga dapat disimpulkan bahwa *REST API backend* yang diimplementasikan mampu memberikan performa yang stabil dan berjalan dengan baik ketika menerima beberapa *request* secara berulang menggunakan *Postman*.



Gambar 20 Tampilan Produk

B. Pembahasan

Berdasarkan hasil pelaksanaan dan pengujian yang telah dilakukan, *REST API backend* pada Sistem Penjualan UMKM Citra Cake Sidamulih berhasil direalisasikan menggunakan *Node.js*, *Express.js*, *PostgreSQL*, dan *Prisma ORM*. Implementasi *backend* menciptakan sejumlah *endpoint* yang mendukung proses autentikasi pengguna, manajemen produk, keranjang belanja, serta pemesanan. Seluruh *endpoint* tersebut dapat diakses melalui protokol *HTTP* menggunakan metode *GET*, *POST*, dan *PATCH* sesuai dengan fungsi masing-masing. Selain itu, *backend* telah berhasil *deploy* pada *Virtual Private Server (VPS)* sehingga *REST API* dapat diakses melalui jaringan internet dan diuji menggunakan aplikasi *Postman*.

Hasil pengujian *Single Track* menunjukkan bahwa seluruh *endpoint* yang diuji memperoleh *Success Rate* sebesar 100% dengan *Status Code* yang sesuai dengan standar *REST API*, yaitu 200 OK dan 201 Created. Hal tersebut menunjukkan bahwa setiap *endpoint* mampu memproses permintaan dengan benar sesuai dengan fungsi yang telah dirancang. Berdasarkan teori *REST API*, *code status HTTP* merupakan indikator keberhasilan komunikasi antara *client* dan *server*. Dengan diperolehnya status kode yang sesuai, dapat disimpulkan bahwa implementasi *endpoint* telah memenuhi prinsip komunikasi *RESTful*.

Pengujian *Multi Track* dilakukan dengan mengirimkan lima kali permintaan pada setiap *endpoint* untuk mengetahui konsistensi performa *backend*. Berdasarkan hasil pengujian, seluruh *endpoint* juga memperoleh *Success Rate* sebesar 100%, sehingga menunjukkan bahwa *backend* mampu memberikan layanan secara konsisten ketika menerima beberapa permintaan secara berulang. Hasil tersebut mengindikasikan bahwa implementasi *REST API* memiliki tingkat keandalan yang baik dalam memproses permintaan pengguna.

BAB V KESIMPULAN

A. Kesimpulan

Berdasarkan hasil pelaksanaan dan evaluasi *REST API backend* pada Sistem Penjualan UMKM Citra Cake Sidamulih menggunakan *Postman*, dapat disimpulkan beberapa hal sebagai berikut.

1. *REST API backend* pada Sistem Penjualan UMKM Citra Cake Sidamulih berhasil diimplementasikan dengan memanfaatkan *Node.js*, *Express.js*, *PostgreSQL*, *Prisma ORM*, *JSON Web Token (JWT)*, *Cloudinary*, dan telah berhasil *deploy* di *Virtual Private Server (VPS)*. Implementasi ini menghasilkan *endpoint REST API* yang mendukung proses autentikasi pengguna, manajemen produk, kategori, keranjang belanja, pemesanan, pembayaran, pengiriman, hingga pemberian ulasan, sehingga mampu mendukung jalannya bisnis sistem penjualan Citra Cake.
2. Hasil dari pengujian menggunakan *Postman* menunjukkan bahwa semua *endpoint REST API* yang diuji berhasil dijalankan sesuai dengan fungsinya. Pada pengujian *Single Track*, semua *endpoint* menghasilkan Status Code yang sesuai dengan standar *REST API*, yaitu 200 OK dan 201 *Created*, serta mencatat *Success Rate* sebesar 100%. Hal ini menandakan bahwa setiap *endpoint* telah beroperasi sesuai dengan kebutuhan sistem. Hasil pengujian *Multi Track* mengungkapkan bahwa *REST API backend* memiliki performa yang stabil ketika menerima lima permintaan (*request*) pada setiap *endpoint*. Semua *endpoint* memperoleh *Success Rate* sebesar 100% dengan rata-rata *Response Time* yang masih dalam batas waktu respons yang baik. Dengan demikian, implementasi *REST API backend* pada Sistem Penjualan UMKM Citra Cake Sidamulih dapat memberikan layanan yang konsisten, handal, dan sesuai dengan tujuan penelitian.

B. Saran

1. Penelitian selanjutnya dapat menambahkan *security testing* dan *performance/load testing* dengan skenario, metrik, dan *tools* khusus agar evaluasi kualitas *REST API* menjadi lebih menyeluruh.
2. Pengembangan sistem selanjutnya dapat memperluas integrasi *frontend*, *payment gateway*, pelacakan pengiriman, dan notifikasi sesuai kebutuhan UMKM.

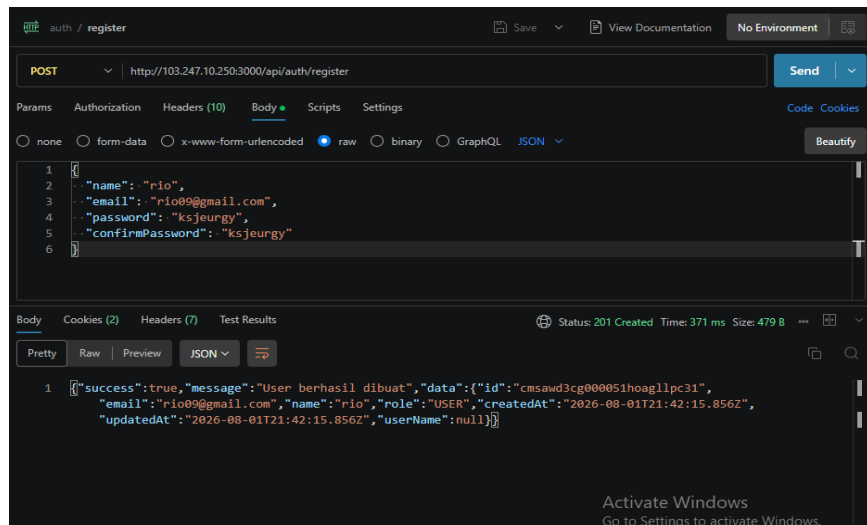
DAFTAR PUSTAKA

- [1] B. Aqila, R. Rabbi, R. G. Guntara, A. Nuryadin, P. Studi, and B. Digital, “Perancangan *Rest Api* untuk Social Media Integration sebagai Layanan Pengelolaan Media Sosial Terpusat bagi Umkm,” 2025. [Online]. Available: <https://ejournal.upi.edu/index.php/IJDB>
- [2] A. W. Prayoga,); Azizah, and N. Fatmawati, “Rancang Bangun Sistem Informasi Penjualan pada Umkm Mj *Sweet* Berbasis *Web*,” 2026.
- [3] S. Minasa, F. Sya’bandyah, M. N. Abdul Muhaemin, and B. Juliandani, “Sistem Informasi Pengelolaan Inventaris Umkm Berbasis Web dengan Pendekatan *Agile*,” *Infotronik : Jurnal Teknologi Informasi dan Elektronika*, vol. 9, no. 2, pp. 104–112, Dec. 2024, doi: 10.32897/infotronik.2024.9.2.3783.
- [4] S. Peldszus *et al.*, “Developer Perspectives On *Rest Api* Usability: A Study Of *Rest Api* Guidelines,” *Companion Proceedings of the 34th ACM Symposium on the Foundations of Software Engineering (FSE ’26)*, June 5â•fi9, 2026, Montreal, Canada, vol. 1, 2026.
- [5] A. Golmohammadi, M. Zhang, and A. Arcuri, “Testing *Restful Apis*: A Survey,” Dec. 2022, [Online]. Available: <http://arxiv.org/abs/2212.14604>
- [6] S. Karlsson, R. Jongeling, A. Čaušević, and D. Sundmark, “Exploring Behaviours Of *Restful Apis* In An Industrial Setting,” *Software Quality Journal*, vol. 32, no. 3, pp. 1287–1324, Sep. 2024, doi: 10.1007/s11219-024-09686-0.
- [7] A. Martin-Lopez, S. Segura, and A. Ruiz-Cortés, “Restest: Automated Black-Box Testing Of *Restful Web Apis*,” in *ISSTA 2021 - Proceedings of the 30th ACM SIGSOFT International Symposium on Software Testing and Analysis*, Association for Computing Machinery, Inc, Jul. 2021, pp. 682–685. doi: 10.1145/3460319.3469082.
- [8] J. Bogner, S. Kotstein, and T. Pfaff, “Do *Restful API* Design Rules Have An Impact On The Understandability Of *Web Apis*?,” *Empir. Softw. Eng.*, vol. 28, no. 6, Nov. 2023, doi: 10.1007/s10664-023-10367-y.
- [9] V. Gonzaga and N. Nawaningtyas, “Implementasi *Rest Api* untuk Sistem Informasi Kepegawaian Berbasis *Spring Boot*,” *Jurnal Nasional Komputasi dan Teknologi Informasi (JNKTI)*, vol. 8, no. 4, 2025.
- [10] H. A. Thooriqoh, B. Mukhamad Mulyo, and A. Rakhmadi, “A Comparative Analysis Of Resource Utilization Using *Iso/Iec 25010* In *Rest Api* File Upload Testing: *Postman* (Gui-Based) And *Cucumber* (Code-Based),” *ILKOMNIKA: Journal of Computer Science and Applied Informatics E-ISSN*, vol. 7, no. 2, pp. 2715–2731, 2025, doi: 10.28926/ilkomnika.v7i2.773.
- [11] A. Poth, O. Rrjolli, and A. Arcuri, “Technology Adoption Performance Evaluation Applied To Testing Industrial *Rest Apis*,” *Automated Software Engineering*, vol. 32, no. 1, Jun. 2025, doi: 10.1007/s10515-024-00477-2.

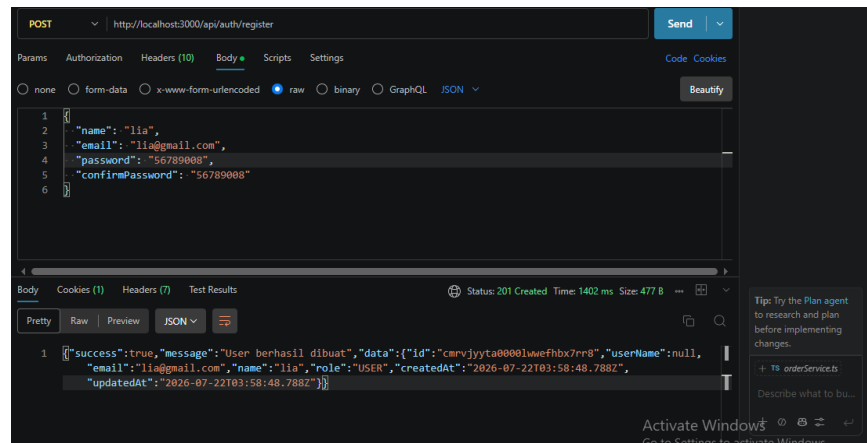
- [12] M. Irsyad, Y. #1, A. #2, and M. Faiqurahman, "JEPIN (Jurnal Edukasi dan Penelitian Informatika) Analisis Perbandingan Efektivitas Arsitektur Restful dan Arsitektur Grpc pada Implementasi Web Service".
- [13] A. Muzakki Aley, A. Irma Purnama Sari, D. Pratama, N. Rahaningsih, and dan Willy Prihartono, "Buletin Sistem Informasi dan Teknologi Islam Analisis Performa *Rest Api* dan *GraphQL* pada Aplikasi Berita Digital di Indonesia *Performance Analysis of REST API and GraphQL in Digital News Applications in Indonesia* Informasi Artikel ABSTRAK," vol. 7, no. 2, pp. 64–70, 2026, doi: 10.33096/busiti.v7i2.3218.
- [14] Abd. Wahab Syahroni, Nindian Puspa Dewi, Nilam Ramadhani, Ubaidi, and Badar Said, "Uji Keamanan *Back End* Aplikasi Berbasis *Website* Menggunakan Metode *Black Box Testing*," *Jurnal PROCESSOR*, vol. 19, no. 2, Oct. 2024, doi: 10.33998/processor.2024.19.2.1752.
- [15] A. W. Jati, A. Pinandito, and N. Y. Setiawan, "Efisiensi Penggunaan *Raw Query* dan *ORM* pada Ekosistem *Node.js*," *Jurna Pengembangan Teknologi Informasi dan Ilmu Komputer*, vol. 9, no. 8, pp. 2548–964, 2025, [Online]. Available: <http://j-ptiik.ub.ac.id>
- [16] Rangga Gelar Guntara and V. Azkarin, "Implementasi dan Pengujian REST API Sistem Reservasi Ruang Rapat dengan Metode *Black Box Testing*," *Jurnal Minfo Polgan*, vol. 12, no. 1, pp. 1229–1238, Jul. 2023, doi: 10.33395/jmp.v12i1.12691.
- [17] I. R. D. Muhammad and I. V. Paputungan, "Development of Backend Server Based on *Rest Api Architecture* in *E-Wallet Transfer System*," *Jurnal Sains, Nalar, dan Aplikasi Teknologi Informasi*, vol. 3, no. 2, pp. 79–87, Jan. 2024, doi: 10.20885/snati.v3.i2.35.
- [18] J. Khatib Sulaiman, R. Syifa Qaisa, A. Putri, H. Maghfirah, and I. Artikel Abstrak, "Perancangan Aplikasi Todo menggunakan *Node.js* dan *REST API*," *Indonesian Journal of Computer Science Attribution*, vol. 12, no. 6, pp. 2023–4132, 2023.
- [19] N. Arif, Amalia, and Misna Asqia, "Pengembangan Sistem Informasi Transaksi Penjualan Umkm Menggunakan Metode *Rapid Application Development*," *The Indonesian Journal of Computer Science*, vol. 14, no. 5, Oct. 2025, doi: 10.33022/ijcs.v14i5.4961.
- [20] M. K. Naufal, F. Affrianto, and A. B. Cahyono, "Implementasi Rest Api untuk Fitur Rencana Strategis Program pada Simpeda," 2022.
- [21] H. A. B. H. Hasanuddin, "Rancang Bangun Rest Api Aplikasi Weshare Sebagai Upaya Mempermudah Pelayanan Donasi Kemanusiaan," 2022.
- [22] T. Listyanto, W. Sri Utami, and U. Teknologi Yogyakarta, "Optimalisasi Aplikasi Toko Bangunan Melalui Integrasi *Rest Api* dan Whatsapp Gateway Optimizing Building Store Applications Through The Integration Of Rest Api And Whatsapp Gateway," *Journal of Information Technology and Computer Science (INTECOMS)*, vol. 7, no. 6, 2024.
- [23] A. Ramadhani, N. Iriadi, and R. Hidayat, "Implementasi Teknologi Rest API Dengan Node Js untuk Aplikasi Rekomendasi Destinasi Wisata," *Journal Computer Science*, vol. 4, no. 1, 2025.

- [24] D. Felicio, J. Simao, and N. Datia, “*Rapitest: Continuous black-box testing of restful web apis*,” in *Procedia Computer Science*, Elsevier B.V., 2023, pp. 537–545. doi: 10.1016/j.procs.2023.01.322.
- [25] E. C. Loquinio, “A Data-Driven Information Kiosk for Enhanced Educational Governance: Integrating Visualization and Geolocation in the DepEd Davao Del Norte Division,” *American Journal of Interdisciplinary Research and Innovation*, vol. 4, no. 4, pp. 33–44, Dec. 2025, doi: 10.54536/ajiri.v4i4.6389.

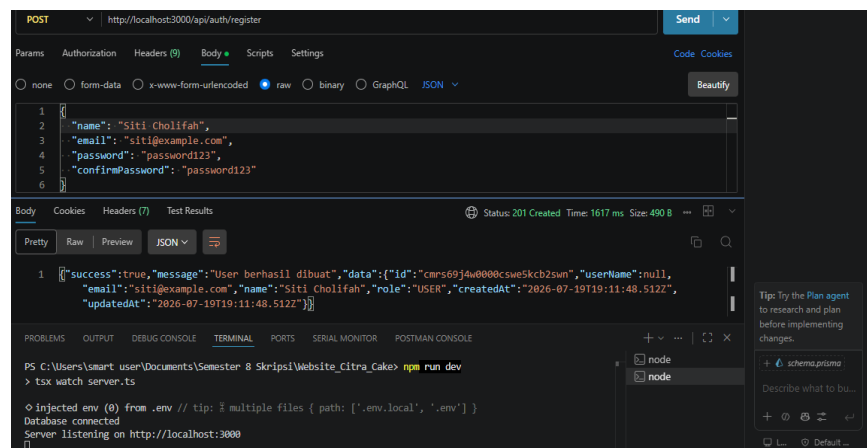
LAMPIRAN



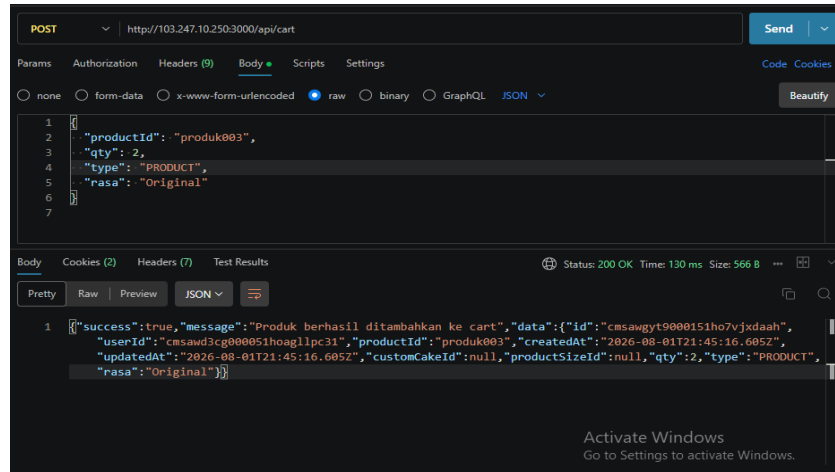
Gambar 21 Register



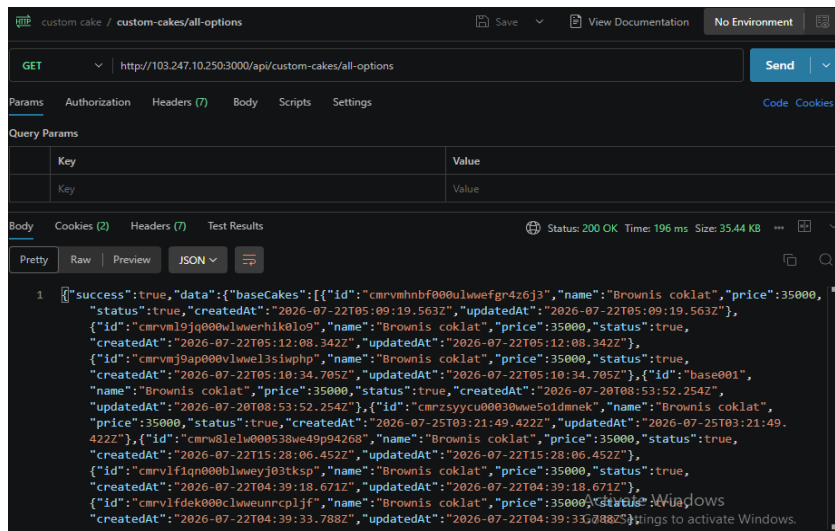
Gambar 22 User



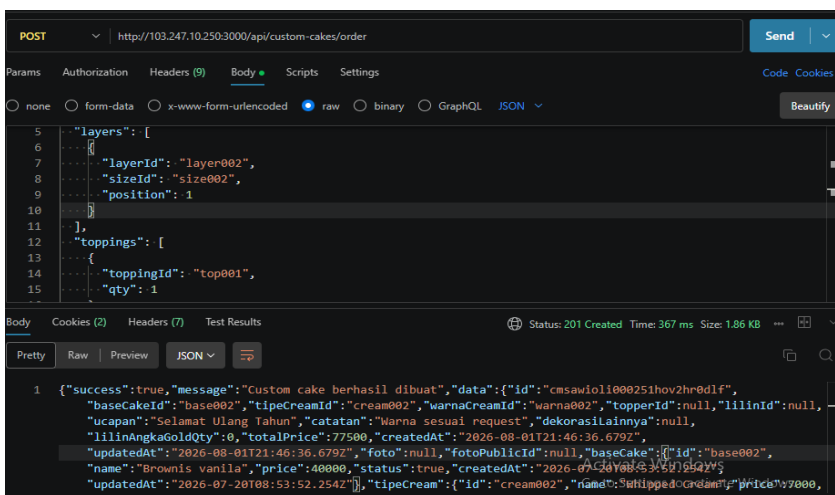
Gambar 23 User



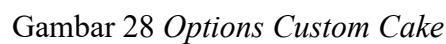
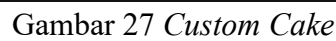
Gambar 24 *Product*

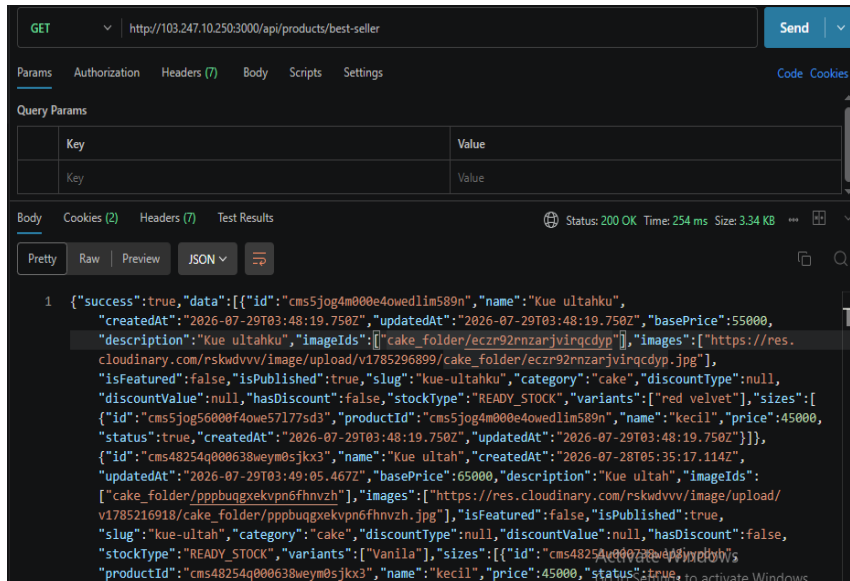


Gambar 25 *Options*

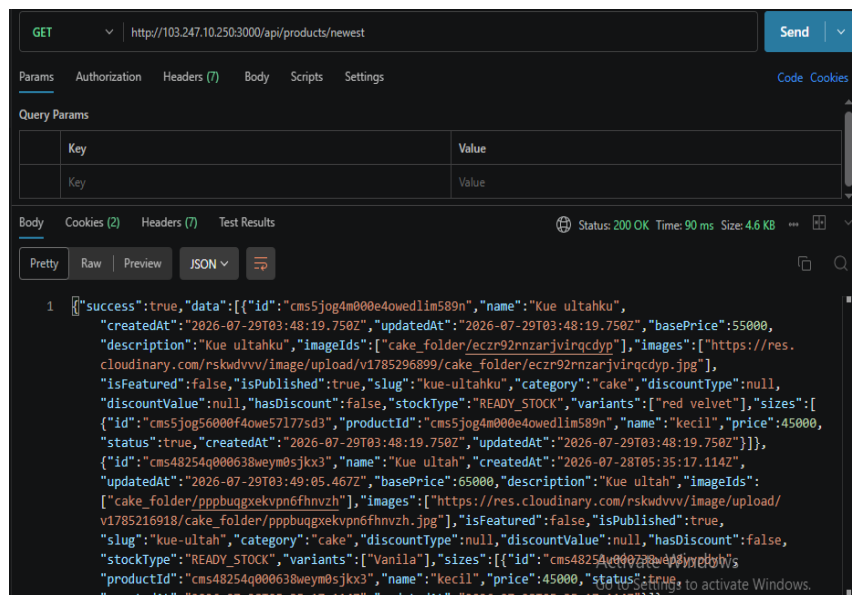


Gambar 26 *Layer Cream*

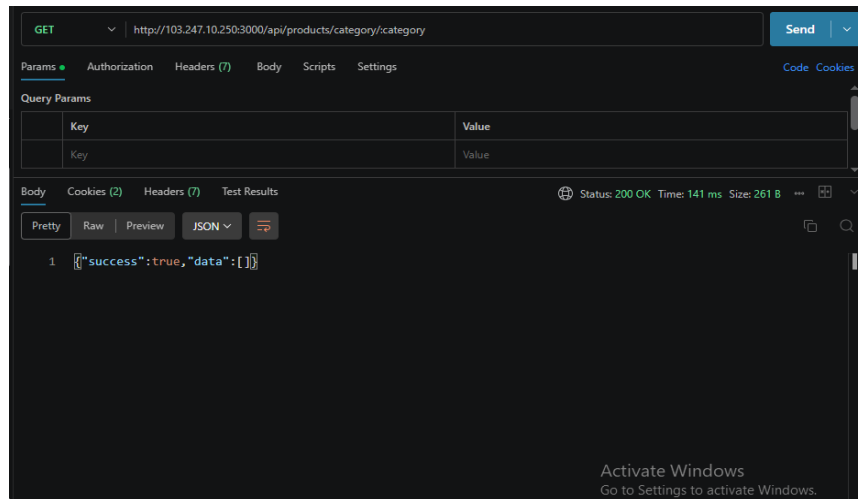




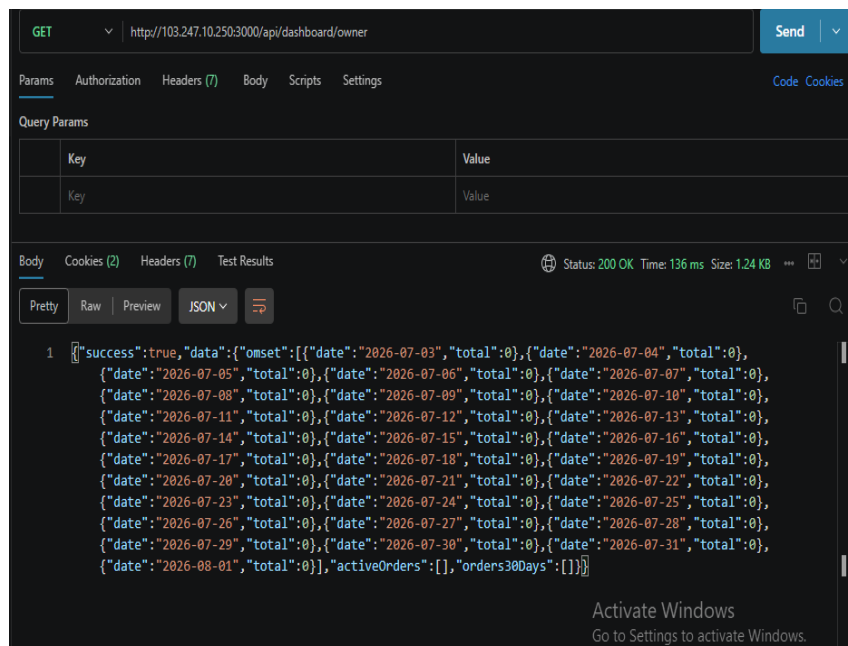
Gambar 30 Best Seller



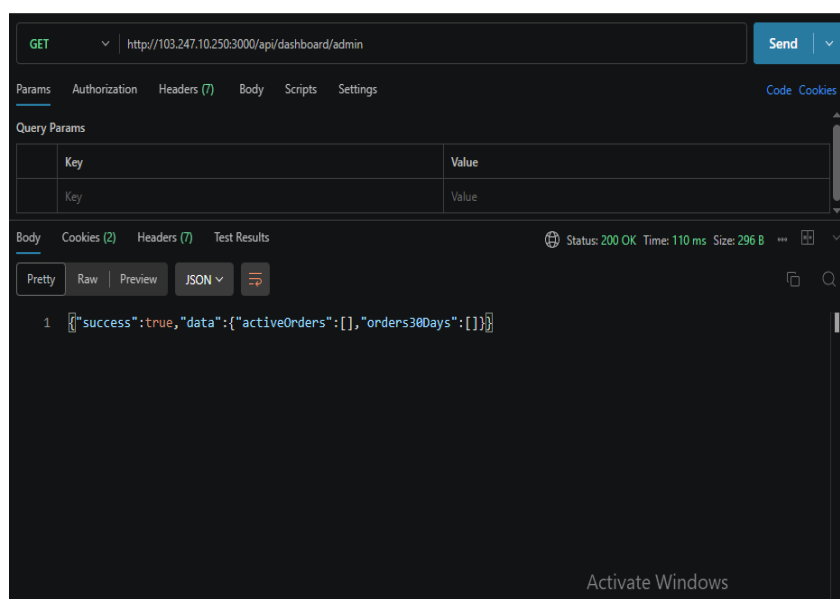
Gambar 31 Newest



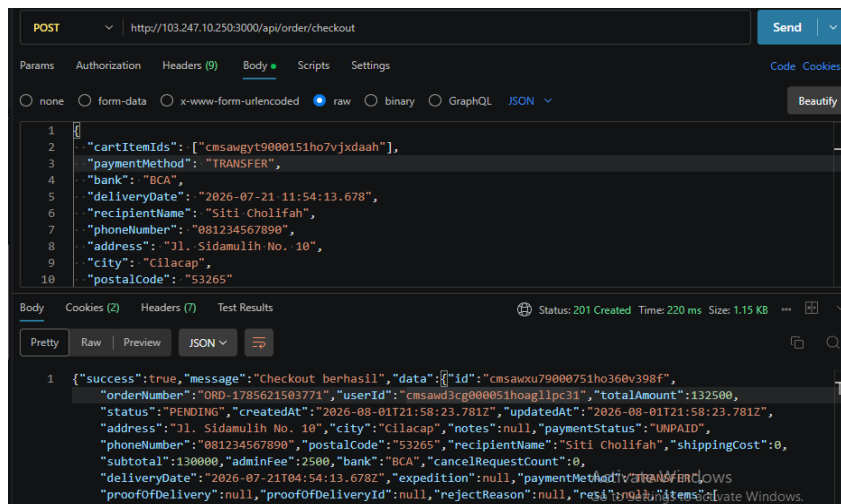
Gambar 32 Sukses *Category*



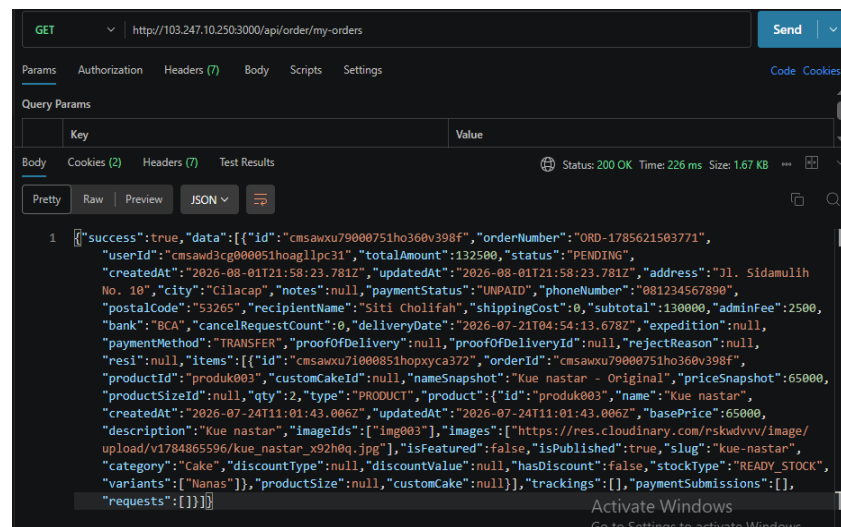
Gambar 33 *Owner*



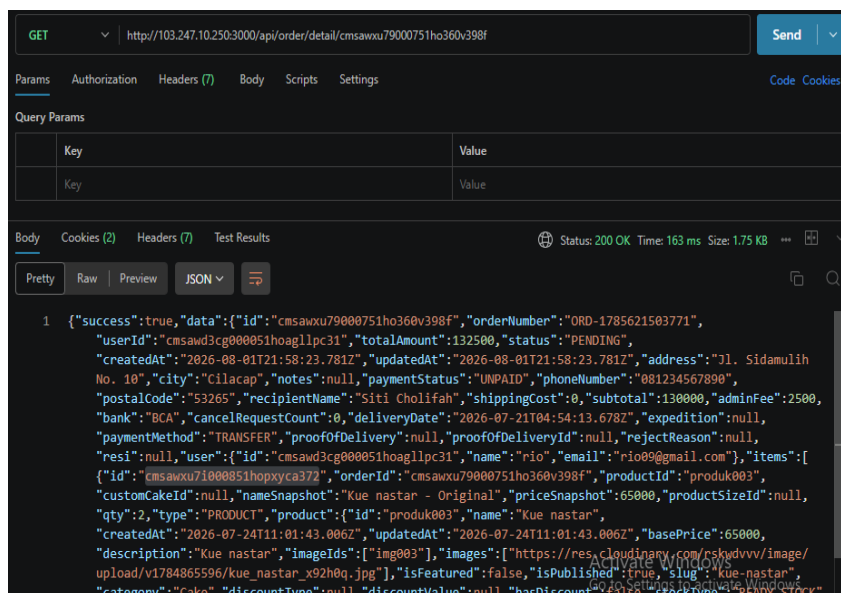
Gambar 34 Sukses data



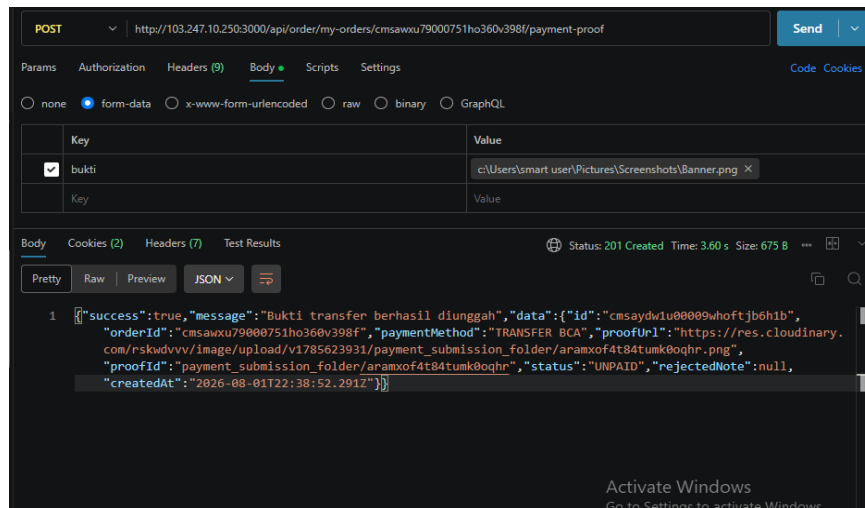
Gambar 35 Checkout



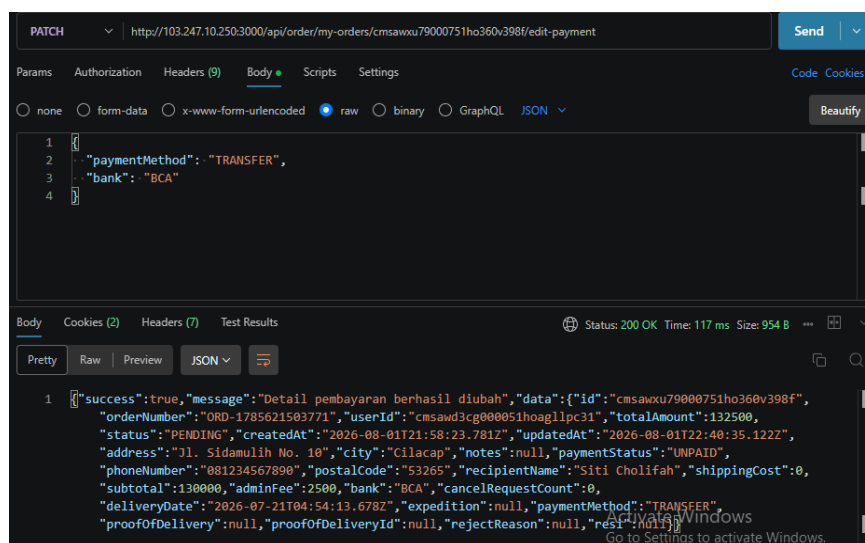
Gambar 36 My Order



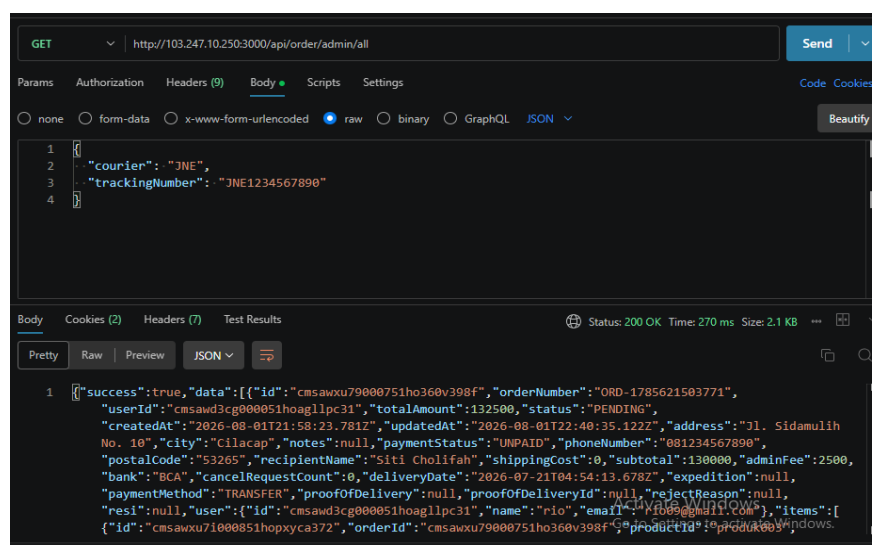
Gambar 37 Sukses Order



Gambar 38 *Payment*



Gambar 39 *Pilihan Payment*



Gambar 40 *Admin*