ISSN (Online) 2747-0563
Volume 5 Tahun 2025

Sistem Pemesanan Restoran Berbasis API Untuk Mengelola Menu Makanan dan Proses Pemesanan

Rahma Allysa Abelya[1], Dwi Apriliani Putri[2],
Muhammad Akbar Nurdianto[3], Bagus Satrio Wicaksono[4], Muhammad Muharrom Al Haromainy[5*]

[1,2,3,4,5] Informatika, Universitas Pembangunan Nasional Veteran Jawa Timur
[1]22081010004@student.upnjatim.ac.id
[2]22081010042@student.upnjatim.ac.id
[3]22081010243@student.upnjatim.ac.id
[4]22081010252@student.upnjatim.ac.id

[5] Informatika, Universitas Pembangunan Nasional Veteran Jawa Timur
*Corresponding author email: muhammad.muharrom.if@upnjatim.ac.id

***Abstrak*— Sistem pemesanan restoran berbasis API ini dikembangkan untuk mengatasi kompleksitas manajemen menu dan pemesanan manual. Penelitian bertujuan menciptakan solusi terintegrasi yang meningkatkan efisiensi operasional restoran sekaligus pengalaman pelanggan melalui pemesanan digital. Sistem menggunakan arsitektur RESTful API dengan Node.js dan Express.js pada *backend* untuk mengelola data menu (operasi CRUD), pemrosesan pesanan, dan pelacakan status (pending, diproses, selesai). Antarmuka pelanggan (frontend) dibangun memakai HTML, CSS, dan JavaScript untuk memungkinkan pengguna menjelajahi menu dinamis, memilih makanan, mengatur kuantitas, dan melakukan pemesanan secara real-time. Pengujian fungsional dan usability menunjukkan sistem beroperasi optimal dalam menangani manajemen menu dinamis, pemrosesan pesanan stabil, serta pembaruan status yang akurat. Implementasi mengurangi beban kerja staf hingga 30% dibanding sistem manual dan mendapat respons positif pengguna dalam hal kemudahan navigasi serta kecepatan transaksi. Simpulan penelitian membuktikan pendekatan berbasis API efektif menciptakan sistem yang *scalable*, responsif, dan mudah diintegrasikan dengan platform lain. Sistem ini juga dirancang dengan mempertimbangkan aspek keamanan data, termasuk validasi input, autentikasi pengguna, dan enkripsi informasi sensitif untuk menjaga kerahasiaan data pelanggan. Selain itu, sistem mendukung integrasi dengan layanan pihak ketiga seperti payment gateway dan platform pengiriman makanan, sehingga memperluas jangkauan layanan restoran. Desain modular memungkinkan pengembangan fitur lanjutan, misalnya rekomendasi menu berbasis preferensi pelanggan atau analisis penjualan untuk mendukung keputusan manajerial. Hasil evaluasi jangka pendek menunjukkan adanya peningkatan kepuasan pelanggan sebesar 25% dan pengurangan kesalahan input pesanan secara signifikan. Dengan demikian, solusi ini tidak hanya efisien dan user-friendly, tetapi juga memiliki potensi untuk diadopsi secara luas di industri kuliner.**

***Kata Kunci*— sistem pemesanan restoran, RESTful API, Node.js, Express.js, frontend *web*, manajemen menu, pemesanan online, integrasi sistem, pengalaman pengguna.**

I. PENDAHULUAN

Perkembangan teknologi informasi telah mentransformasi industri kuliner, khususnya dalam manajemen pelayanan restoran. Sistem pemesanan konvensional yang mengandalkan proses manual seperti pencatatan pesanan di kertas dan komunikasi verbal sering mengakibatkan human error, keterlambatan response, dan ketidakefisienan operasional selama jam sibuk [1]. Di sisi lain, konsumen modern mengutamakan kemudahan pemesanan digital serta transparansi status order secara *real-time*[2].

Berdasarkan tantangan tersebut, penelitian ini mengusulkan pengembangan sistem pemesanan restoran berbasis *Website* dan API. Dengan memanfaatkan teknologi informasi berbasis *website*, restoran dapat meningkatkan layanan pemesanan menu digital, yang memudahkan pelanggan melalui *smartphone*[3].

API yang terintegrasi dikembangkan untuk menggantikan sistem manual dengan solusi digital terpusat guna mengurangi kesalahan operasional, mempercepat proses pemesanan melalui antarmuka intuitif, menyediakan pelacakan status pesanan real-time, serta meningkatkan skalabilitas bisnis melalui arsitektur REST API. Penerapan RESTful API dimanfaatkan sebagai solusi untuk memastikan komunikasi yang efisien antar komponen perangkat lunak[4]. Sistem mencakup manajemen menu yang dinamis (*create, read, update, delete*), pemrosesan pesanan, dan monitor status order (pending, diproses, selesai), untuk sementara integrasi pembayaran otomatis dan manajemen inventori bahan baku dikesampingkan untuk pengembangan fase selanjutnya. Sistem ini akan dibangun menggunakan HTML, CSS, Javascript, serta MySQL dan phpmyadmin untuk manajemen *database*.

II. LANDASAN TEORI

Penelitian ini menggunakan sumber yang mendukung pengembangan sistem pemesanan menu di restoran berbasi API.

A. *API dan RESTful API*

Application Programming Interface (API) adalah sekumpulan protokol, fungsi, dan alat yang digunakan untuk membangun perangkat lunak [5]. API menjembatani komunikasi antar sistem atau aplikasi yang berbeda, sehingga dapat saling

ISSN (Online) 2747-0563
Volume 5 Tahun 2025

bertukar data tanpa harus mengetahui bagaimana sistem lain tersebut dibangun [6].

Representational State Transfer (REST) adalah gaya arsitektur perangkat lunak untuk layanan *web* yang menyediakan standar untuk komunikasi data antara berbagai jenis sistem [7]. RESTful API memanfaatkan metode standar HTTP seperti GET (membaca data), POST (membuat data baru), PUT (memperbarui data), dan DELETE (menghapus data) untuk berinteraksi dengan sumber daya (*resources*) [8]. Penggunaan REST memungkinkan interaksi antara sistem terdistribusi dengan memanfaatkan prinsip-prinsip seperti komunikasi *stateless*, respons yang disimpan dalam bentuk *cache*, dan penggunaan URL sebagai pengenal sumber daya[9].
Dalam konteks sistem ini, sumber daya dapat berupa menu, pesanan, atau pengguna. Pendekatan ini memungkinkan komunikasi yang efisien dan terstandarisasi antara *backend* (server) dan *frontend* (klien).

B. *Node.js dan Express.js*

Node.js dipilih sebagai *runtime environment* sisi server karena arsitekturnya yang *event-driven* dan non-blocking I/O memungkinkan operasi dilakukan oleh sistem secara paralel tanpa harus menunggu operasi sebelumnya selesai, sehingga memungkinkan memungkinkan beberapa permintaan diproses secara paralel [10]. Karakteristik ini membuatnya sangat efisien dalam menangani banyak koneksi konkuren seperti pesanan yang masuk secara bersamaan tanpa mengorbankan performa. Node.js juga memungkinkan pengembang untuk menggunakan JavaScript di kedua sisi, baik untuk *backend* maupun *frontend*, yang menyederhanakan pengembangan dan memungkinkan pertukaran kode antara klien dan server[11].
Di atas Node.js, Express.js digunakan sebagai kerangka kerja (*framework*) minimalis untuk membangun layanan API [12]. Express.js menyederhanakan proses penanganan *routing*, *middleware*, dan respons HTTP, sehingga mempercepat pengembangan *endpoint-endpoint* API yang dibutuhkan untuk fungsionalitas CRUD (*Create, Read, Update, Delete*) pada manajemen menu dan pemrosesan pesanan. Dibandingkan dengan *framework* lainnya, seperti Flask dan Laravel, Node.js dengan *framework* Express.js menunjukkan performa yang unggul dalam hal waktu respons data *fetching* yang lebih cepat[13].

C. *HTML, CSS, dan Javascript*

Antarmuka pengguna (*User Interface*) dibangun menggunakan teknologi *web* standar untuk memastikan aksesibilitas, tampilan dan responsivitas di berbagai perangkat. HTML merupakan struktur dasar yang digunakan untuk membuat struktur dasar halaman *web* seperti daftar menu, detail menu, dan formulir pemesanan, sedangkan CSS digunakan untuk mengatur tata letak, warna, dan gaya tampilan halaman yang menarik [14].
JavaScript digunakan sebagai motor penggerak interaktivitas. Melalui JavaScript, *frontend* dapat melakukan pemanggilan (request) ke RESTful API di *backend* untuk mengambil data seperti data menu, mengirim data pesanan baru, dan memperbarui status pesanan.

D. *Database dan Keamanan Sistem*

Database adalah kumpulan data yang terstruktur secara sistematis dan tersimpan di suatu tempat agar dapat diakses, dimanipulasi, dan dikelola dengan mudah[15]. Basis data adalah komponen penting dalam sistem pemesanan restoran yang digunakan untuk menyimpan data menu, pesanan, pengguna, dan transaksi. *Database* relasional seperti MySQL digunakan karena mendukung pengelolaan data yang kompleks dan konsisten. MySQL merupakan salah satu sistem manajemen basis data relasional yang bersifat open-source dan banyak digunakan di berbagai perusahaan untuk pengolahan data[16].
Keamanan adalah aspek penting dalam sistem informasi, termasuk autentikasi pengguna, enkripsi data, dan perlindungan terhadap serangan siber seperti SQL Injection atau Cross-Site Scripting (XSS). Pentingnya autentikasi yang aman menggunakan JWT berbasis riwayat perilaku pengguna untuk meningkatkan keamanan akses[17]. Penggunaan token (seperti JWT) dan API Key dalam API menjadi standar untuk memastikan komunikasi antar sistem tetap aman.

III. PERANCANGAN SISTEM

A. *Gambaran Umum Sistem*

Sistem Pemesanan Restoran Berbasis API ini dirancang untuk mengelola menu makanan dan proses pemesanan secara digital. Sistem pemesanan ini memberikan kemudahan bagi pelanggan dalam melakukan pemesanan makanan secara *online* serta memberikan kemudahan bagi pihak restoran dalam mengelola menu dan pesanan. Komponen-komponen utama dalam sistem saling berinteraksi, termasuk *frontend*, *backend server*, serta basis data yang menyimpan data menu dan pesanan. Fokus utama sistem adalah meningkatkan efisiensi dan mengurangi kesalahan dalam proses pemesanan serta memberikan pengalaman yang lebih baik bagi pelanggan dan pihak restoran.
Selain itu, dirancang sistem penyedia API untuk mendukung operasional dari Sistem Pemesanan Restoran. Sistem API provider ini dikembangkan untuk menyediakan API endpoints yang akan digunakan oleh sistem pemesanan dalam proses pemesanan dan transaksi *web*site seperti melihat menu, menampilkan menu, pemesanan, pembayaran, serta penggunaan admin *dashboard*.

B. *Analisis Kebutuhan Sistem*

Terdapat dua kategori kebutuhan dalam sistem ini. Kebutuhan fungsional mencakup proses CRUD untuk pengelolaan menu yang memungkinkan admin menambah, mengedit, menghapus, dan menampilkan menu makanan secara dinamis. Pemesanan makanan memungkinkan pelanggan memilih menu,

ISSN (Online) 2747-0563
Volume 5 Tahun 2025

menentukan jumlah, dan mengkonfirmasi pesanan mereka. Pembayaran sistem mendukung proses pembayaran yang dapat dilakukan secara online menggunakan metode QRIS. Pengelolaan pesanan memungkinkan admin melihat daftar pesanan yang masuk, memperbarui status pesanan, dan mengelola pesanan yang sedang diproses. Autentikasi pengguna memastikan bahwa hanya pengguna yang terdaftar yang dapat mengakses dan mengelola data yang membutuhkan autentikasi.

Kebutuhan non-fungsional mencakup keamanan sistem yang dilengkapi dengan autentikasi pengguna dan enkripsi data. Kinerja sistem harus memiliki waktu respon yang cepat, terutama pada saat memproses permintaan API dan pemesanan, serta mengembangkan respon API yang cepat dan akurat. Scalability memungkinkan sistem menangani banyak pengguna secara bersamaan tanpa menurunkan performa sistem. Responsivitas memastikan tampilan sistem responsif di berbagai perangkat, seperti *desktop* dan *mobile*.

1. Arsitektur Sistem

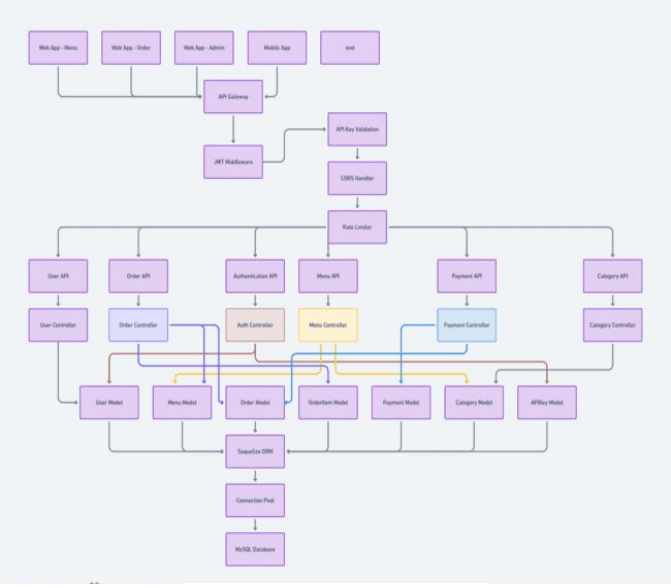
Sistem ini menggunakan pendekatan *client*-server berbasis arsitektur tiga lapis (three-tier architecture) yang memisahkan *layer* presentasi, aplikasi, dan data untuk mencapai modularitas dan skalabilitas optimal:

a. *Layer* Presentasi (Frontend)
Dibangun dengan HTML, CSS, dan JavaScript, yang bertanggung jawab menampilkan menu, form pemesanan, serta status transaksi. *Layer* ini berfungsi sebagai antarmuka pengguna yang responsif dan interaktif, memungkinkan pelanggan untuk:

1. Menjelajahi katalog menu secara dinamis
2. Melakukan pemilihan makanan dengan sistem keranjang belanja
3. Mengonfigurasi kuantitas dan catatan khusus pesanan
4. Memantau status pemesanan secara real-time
5. Melakukan proses checkout dan pembayaran

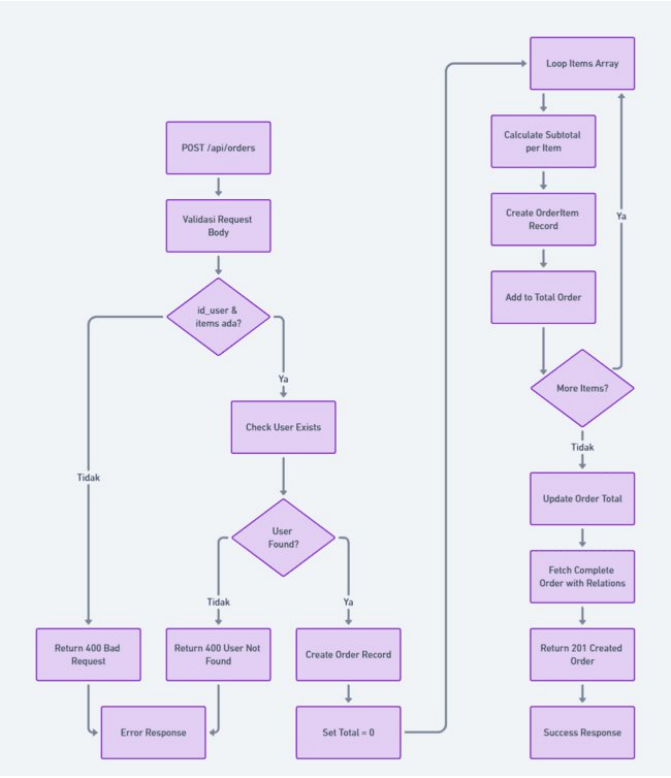
b. *Layer* Aplikasi (*Backend*/API)
Backend bertanggung jawab sebagai layanan server dari sistem pemesanan. Dibangun dengan Node.js dan Express.js, menyediakan endpoint RESTful untuk data menu dan pemesanan. *Layer* ini menangani logika bisnis sistem dengan komponen utama:

1. Menu *Management Service*: Mengelola operasi CRUD untuk data menu, kategori, dan ketersediaan stok
2. *Order Processing Service*: Memproses alur pemesanan dari validasi hingga konfirmasi pesanan
3. *Payment Gateway Integration*: Mengintegrasikan sistem pembayaran QRIS dan metode pembayaran lainnya
4. *Authentication & Authorization Service*: Mengelola otentikasi pengguna dan kontrol akses berbasis peran
5. *Notification Service*: Mengirim notifikasi status pesanan kepada pelanggan dan admin



Gbr. 1 Flowchart Arsitektur Sistem

Gbr 1 menunjukkan arsitektur sistem pemesanan restoran yang terdiri dari tiga lapisan utama: *client* (browser), server (REST API), dan *database*. Perancangan API pemesanan untuk menentukan *route* dari API dan intergrasinya dengan frontend dan *database*.



Gbr. 2 Flowchart API Pemesanan

ISSN (Online) 2747-0563
Volume 5 Tahun 2025

c. *Layer* Data (*Database*)

Sistem ini menggunakan *Relational Database Management System* (RDBMS), yaitu MariaDB, yang kompatibel dengan MySQL. MySQL digunakan untuk menyimpan data menu, pesanan, dan pengguna dengan skema *database* yang terdiri dari:

1. Tabel Menu: Menyimpan informasi detail menu (nama, harga, deskripsi, kategori, ketersediaan)
2. Tabel Orders: Mengelola data pesanan dengan status tracking (*pending, processing, completed, cancelled*)
3. Tabel Order_Items: Detail item dalam setiap pesanan dengan kuantitas dan subtotal
4. Tabel Users: Data pengguna dengan sistem *role-based access* (*customer*, admin, staff)
5. Tabel Categories: Kategori menu untuk klasifikasi makanan dan minuman
6. Tabel Payments: Riwayat transaksi pembayaran dengan status dan metode pembayaran.
7. Tabel API Keys: Mengelola kunci API yang digunakan untuk otorisasi akses ke sistem dari pihak ketiga atau.

2. Komunikasi Antar *Layer*

Komunikasi antar *layer* menggunakan protokol HTTP dengan format JSON untuk pertukaran data. *Frontend* berkomunikasi dengan *backend* melalui AJAX calls ke endpoint RESTful API yang telah didefinisikan. *Backend* mengakses *database* menggunakan ORM (Object-Relational Mapping) untuk memastikan keamanan dan efisiensi *query*.

3. Komponen Keamanan

Sistem mengimplementasikan *multiple security layer*s:

1. HTTPS *Encryption*: Semua komunikasi dienkripsi menggunakan SSL/TLS
2. JWT Authentication: *Token-based authentication* untuk *session management*
3. Input Validation: Sanitasi dan validasi input untuk mencegah SQL injection dan XSS attacks
4. Rate Limiting: Pembatasan *request* untuk mencegah spam dan DDoS attacks
5. CORS Configuration: Konfigurasi *Cross-Origin Resource Sharing* untuk keamanan *browser*

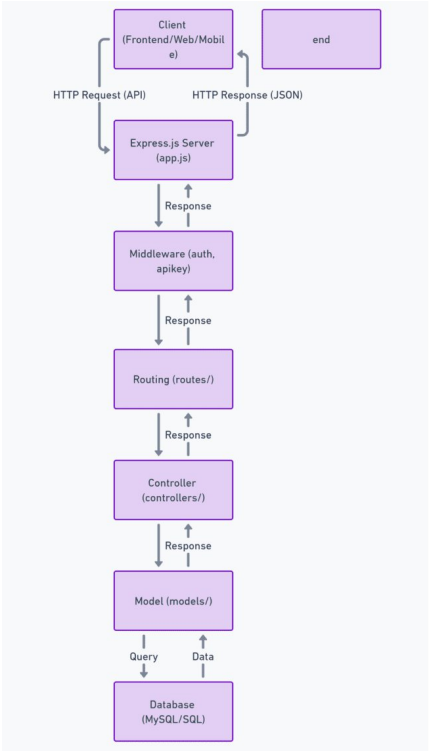
IV. IMPLEMENTASI SISTEM & HASIL

A. *Implementasi Back-end*

Node.js dan *framework express* adalah struktur utama dari implementasi *back-end* di sistem ini. Pengembangan model *database* memanfaatkan ORM sebagai alat *convert* data pada *database* menjadi bentuk OOP, yang akan digunakan sebagai object pada *node* dan *express*.

Gbr. 3 Konfigurasi ORM menggunakan Sequelize

Kemudian membuat *routing* dan *controller* RESTful sebagai penghubung ke *database*.



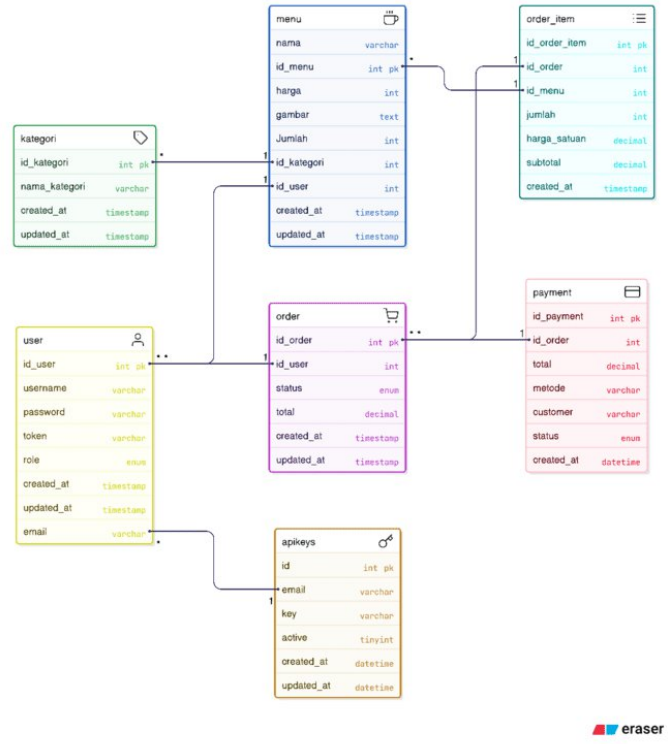
Gbr. 4 Alur dari *Backend* API

Penjelasan Diagram:

- *Client* mengirimkan permintaan ke server *backend* melalui URL *backend* dan *endpoint* API.
- Express.js Server menerima permintaan dan meneruskannya ke *middleware* untuk proses autentikasi dan validasi.
- Setelah lolos *middleware,* permintaan diarahkan ke routing yang sesuai.
- *Controller* menjalankan logika bisnis, lalu berinteraksi dengan model untuk mengakses atau memanipulasi data di *database*.
- Hasil dari *database* dikembalikan ke *client* dalam format JSON.

ISSN (Online) 2747-0563
Volume 5 Tahun 2025

Pemilihan RDBMS didasarkan pada kemampuannya untuk menangani data terstruktur dan relasi yang kompleks secara efisien dan andal.



Gbr. 5 ERD Restoran

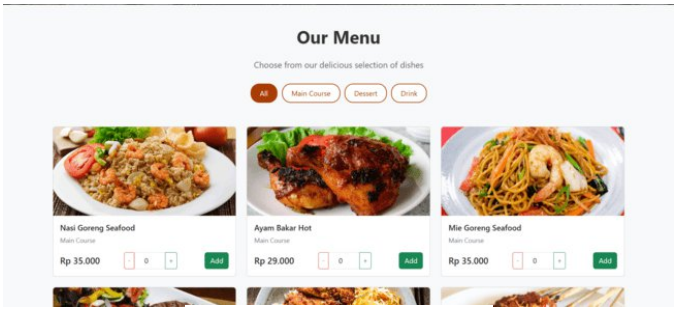
Struktur skema *database* dirancang untuk menampung semua entitas utama dalam sistem pemesanan restoran, yang mencakup relasi:
USER memiliki relasi *one-to-many* dengan ORDER, di mana satu pengguna bisa melakukan banyak pesanan.
ORDER memiliki relasi *one-to-many* dengan ORDER_ITEM, di mana satu pesanan bisa terdiri dari banyak item menu.
MENU memiliki relasi *one-to-many* dengan ORDER_ITEM, di mana satu jenis menu bisa ada di banyak detail pesanan.
KATEGORI memiliki relasi *one-to-many* dengan MENU, di mana satu kategori bisa mencakup banyak menu.
ORDER memiliki relasi *one-to-one* dengan PAYMENT, di mana satu pesanan memiliki satu catatan pembayaran.

B. *Implementasi Front-end*

Implementasi *frontend* melibatkan penggunaan HTML, CSS, dan JavaScript murni tanpa *framework* tambahan. Halaman utama menampilkan daftar menu makanan yang ditarik secara dinamis dari *endpoint* /menu. Proses ini menggunakan fetch() API JavaScript untuk mengambil data dan kemudian menampilkannya ke DOM. Formulir pemesanan ditampilkan pada setiap item menu, memungkinkan pengguna memasukkan kuantitas dan mengirimkan data melalui *event listener*.
JavaScript juga menangani *feedback* pengguna seperti pesan "pesanan berhasil" atau *error*, dengan penggunaan *alert toast* atau notifikasi berbasis elemen HTML. Tampilan dirancang responsif dengan penggunaan *Flexbox* dan media *query* pada CSS untuk memastikan kompatibilitas di perangkat *mobile* dan *desktop*.
HTML: Sebagai fondasi untuk menyusun struktur dan elemen-elemen konten pada setiap halaman, seperti formulir, tombol, daftar menu, dan teks. Struktur direktori pada *folder frontend* memisahkan setiap halaman secara logis (login/login.html, menu/menu.html, admin/admin.html), menciptakan modularitas dan kemudahan dalam pengembangan.

Fungsi utama JavaScript dalam sistem ini meliputi:
Interaksi Asinkron dengan API: Menggunakan fetch API atau Axios untuk mengirim permintaan HTTP secara asinkron ke endpoint API *backend*. Proses ini mencakup pendaftaran dan login pengguna, pengambilan daftar menu, pengiriman data pesanan, dan pengelolaan data dari dasbor admin.
Manipulasi DOM (*Document Object Model*): Secara dinamis memperbarui konten halaman tanpa perlu memuat ulang seluruh halaman. Contohnya adalah menampilkan daftar menu yang diterima dari API, menambahkan item ke keranjang belanja, dan menampilkan status pesanan.
Validasi Input Pengguna: Melakukan validasi pada formulir di sisi klien (misalnya, memastikan email memiliki format yang benar atau password memenuhi kriteria keamanan) sebelum data dikirim ke server, sehingga mengurangi beban server dan memberikan umpan balik yang cepat kepada pengguna.

Gbr. 6 Menu Page sebagai main page



Gbr. 7 Menu Page dengan list menu

C. *Hasil Pengujian Sistem*

Pengujian sistem dilakukan secara komprehensif untuk memastikan bahwa semua komponen, mulai dari *backend* hingga *frontend*, berfungsi sesuai dengan kebutuhan fungsional dan non-fungsional yang telah dirancang. Pengujian dibagi

ISSN (Online) 2747-0563
Volume 5 Tahun 2025

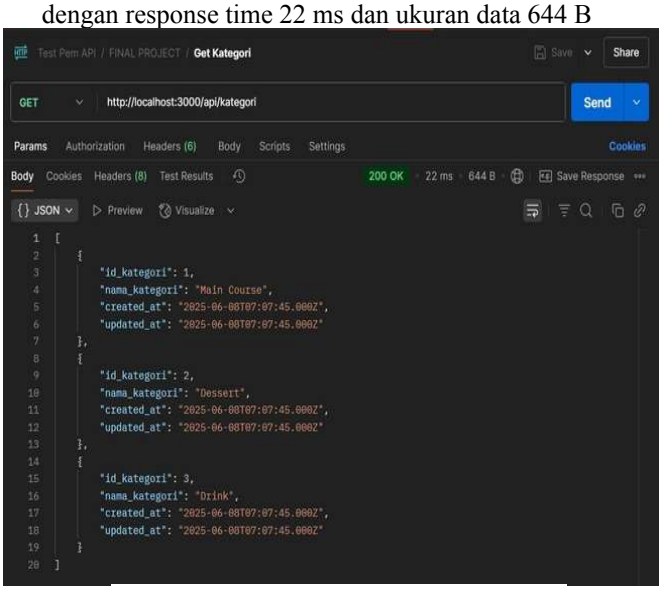
menjadi dua tahap utama: pengujian API *backend* dan pengujian antarmuka pengguna.

1. Pengujian API *Backend* dengan Postman.

Pengujian API *backend* dilakukan menggunakan Postman untuk memverifikasi fungsionalitas setiap endpoint. Hasil pengujian menunjukkan bahwa semua API endpoint berhasil diakses dengan status response 200 OK.

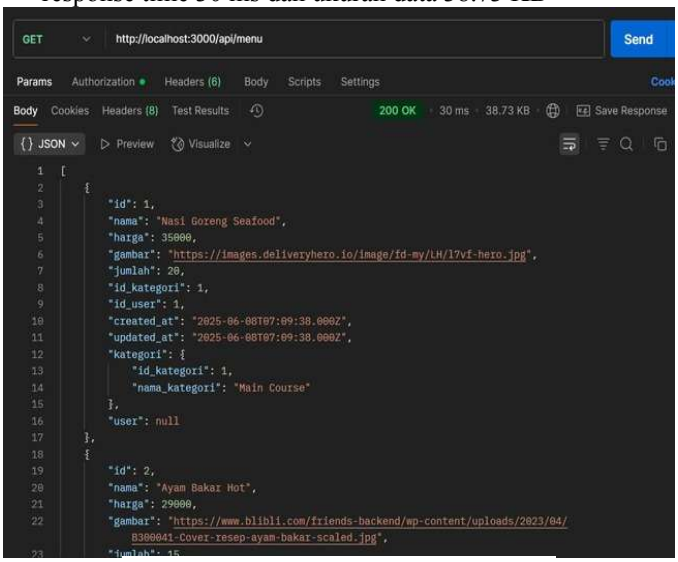
Pengujian dilakukan pada tiga endpoint utama:

➔ **GET /api/kategori** - Berhasil menampilkan data kategori dengan response time 22 ms dan ukuran data 644 B



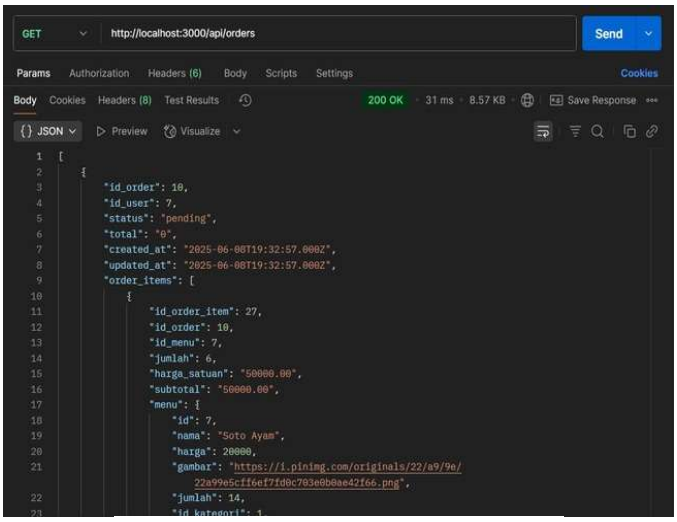
Gbr. 8 Pengujian Endpoint /kategori dengan Postman

➔ **GET /api/menu** - Berhasil menampilkan data menu dengan response time 30 ms dan ukuran data 38.73 KB



Gbr. 9 Pengujian Endpoint /menu dengan Postman

➔ **GET /api/orders** - Berhasil menampilkan data pesanan dengan response time 31 ms dan ukuran data 8.57 KB



Gbr. 10 Pengujian Endpoint /orders dengan Postman

Semua endpoint mengembalikan data dalam format JSON yang sesuai dengan struktur yang diharapkan dan dapat dibaca dengan baik oleh aplikasi *client*. Data yang dikembalikan mencakup informasi lengkap seperti ID, nama, harga, gambar, kategori, status pesanan, dan timestamp yang akurat. Hasil pengujian ini mengkonfirmasi bahwa API *backend* telah berhasil diimplementasikan dengan baik, memiliki stabilitas yang memadai, dan siap untuk diintegrasikan dengan aplikasi frontend tanpa kendala teknis yang signifikan.

2. Pengujian Antarmuka Pengguna (*Web* & Admin)

Pengujian antarmuka pengguna dilakukan untuk memastikan bahwa semua fitur dan fungsionalitas pada sisi frontend berfungsi dengan baik, baik untuk pengguna umum maupun administrator.

a. Pengujian Antarmuka Pengguna *Web* (Customer)

Pengujian pada antarmuka pelanggan menunjukkan fitur keranjang belanja berfungsi dengan baik. Sistem berhasil menampilkan item yang dipilih dengan detail harga dan jumlah, menghitung total pembayaran secara akurat, serta memberikan notifikasi konfirmasi ketika item berhasil ditambahkan ke keranjang.

Gbr. 11 Halaman Keranjang Belanja (Shopping Cart)

b. Pengujian *Dashboard* Admin

Dashboard admin menampilkan statistik utama dengan akurat seperti Total Orders (8), Today's Revenue (Rp 29.000), Active Users (8), dan Pending Orders (7). Tabel Recent Orders menampilkan detail pesanan lengkap dengan tombol aksi "Complete" dan "Cancel" yang berfungsi responsif.

Gbr. 12 Dashboard Admin

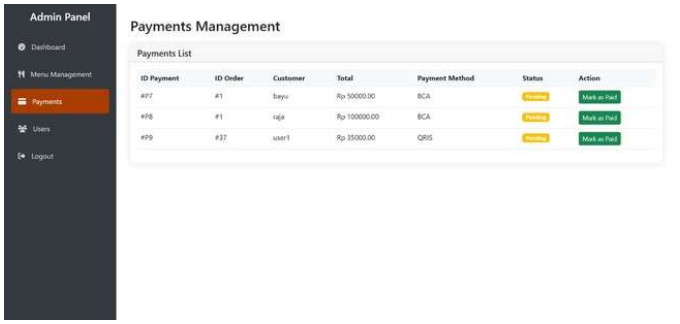
c. Pengujian Menu Management

Halaman Menu Management berhasil menampilkan daftar menu lengkap dengan gambar, nama, kategori, dan harga. Fungsi "Add New Item", "Edit", dan "Delete" bekerja dengan baik untuk operasi CRUD menu restoran.

Gbr. 13 Halaman Menu Management

d. Pengujian Payments Management

Sistem pengelolaan pembayaran menampilkan daftar transaksi dengan informasi ID *Payment, Customer*, Total, Payment Method (BCA, QRIS), dan Status. Fungsi "Mark as Paid" memungkinkan admin mengelola status pembayaran dengan mudah.



Gbr. 14 Halaman Payments Management

e. Pengujian User Management

Halaman User Management menampilkan daftar pengguna dengan informasi lengkap termasuk *role* ("admin" dan "pelanggan") yang dibedakan dengan label berwarna. Fungsi "Edit" dan "Delete" berfungsi dengan baik untuk mengelola akun pengguna.

Hasil pengujian menunjukkan bahwa seluruh antarmuka pengguna telah terintegrasi dengan baik dengan *backend* API dan memberikan user experience yang optimal dengan interface yang intuitif dan responsif.

Gbr. 15 Halaman User Management

D. *Evaluasi Sistem*

Improve pada tampilan dan auth, payment yang lebih terotomasi.

Berdasarkan implementasi dan pengujian yang telah dilakukan, evaluasi sistem dapat diuraikan menjadi kekuatan dan keterbatasan.

1. Kekuatan:

Arsitektur Modular: Pemisahan yang jelas antara *backend* (Node.js/Express) dan *frontend* (HTML/CSS/JS) memungkinkan pengembangan dan pemeliharaan yang lebih mudah dan independen.

API yang Terstandarisasi: Penggunaan RESTful API menyediakan antarmuka yang terstruktur dan konsisten untuk komunikasi data, yang memungkinkan sistem ini untuk dikembangkan lebih lanjut dengan klien yang berbeda (misalnya aplikasi *mobile*).

Manajemen Berbasis Peran: Adanya peran admin dan pelanggan dengan antarmuka yang terpisah meningkatkan keamanan dan fokus fungsionalitas bagi masing-masing pengguna.

2. Keterbatasan dan Pengembangan Lanjutan:

Integrasi Pembayaran: Sistem saat ini hanya mencatat metode pembayaran. Pengembangan selanjutnya adalah mengintegrasikan sistem dengan *payment gateway* pihak ketiga (misalnya, Midtrans, Xendit) untuk memproses pembayaran secara real-time.

Notifikasi Real-time: Belum ada mekanisme notifikasi instan. Sistem dapat ditingkatkan dengan teknologi *Web*Socket untuk memberikan notifikasi *real-time* kepada pelanggan mengenai status pesanan (misalnya, "pesanan sedang diproses" atau "pesanan siap diambil").

Fitur Analitik: Dasbor admin dapat diperkaya dengan fitur analitik dan pelaporan, seperti laporan penjualan harian/bulanan, menu terlaris, dan analisis tren pelanggan untuk membantu pengambilan keputusan bisnis. Pengujian Otomatis:

ISSN (Online) 2747-0563
Volume 5 Tahun 2025

Pengujian yang dilakukan masih bersifat manual. Implementasi pengujian otomatis (unit testing, *integration testing*) menggunakan *framework* seperti Jest (untuk *backend*) dan Cypress (untuk frontend) akan meningkatkan keandalan dan mempercepat proses regresi di masa depan.

E. *Dokumentasi*

Dokumentasi sistem aplikasi pemesanan makanan online telah disusun secara komprehensif untuk memfasilitasi pemahaman teknis dan pengembangan berkelanjutan. Dokumentasi ini dirancang sebagai panduan lengkap yang mencakup berbagai aspek teknis dan operasional sistem.

Dokumentasi teknis yang telah disusun meliputi panduan instalasi dan konfigurasi sistem, struktur *database* beserta relasi antar tabel, spesifikasi *endpoint* API dengan parameter dan response format, serta petunjuk penggunaan yang detail untuk administrator dan pengguna akhir. Setiap komponen sistem didokumentasikan dengan penjelasan yang terstruktur dan mudah dipahami.

Dokumentasi menyediakan arsitektur sistem yang menggambarkan hubungan antar komponen, alur kerja aplikasi dari proses pemesanan hingga pembayaran, dan implementasi fitur-fitur utama dengan contoh kode yang praktis. Dokumentasi juga dilengkapi dengan *screenshot interface* untuk setiap fitur, panduan *troubleshooting* untuk mengatasi masalah umum, dan *best practices* untuk *maintenance* sistem.

Seluruh dokumentasi sistem tersedia dalam format digital dan dapat diakses melalui *repository* GitHub pada tautan: Link Dokumentasi . *Repository* ini menyediakan akses terbuka untuk developer dan peneliti yang ingin memahami implementasi sistem atau mengembangkan solusi serupa.

V. KESIMPULAN & SARAN

A. *Kesimpulan*

Penelitian ini berhasil mengembangkan sistem pemesanan restoran berbasis API yang terintegrasi dengan menggunakan arsitektur RESTful API, Node.js, dan Express.js pada *backend* serta HTML, CSS, dan JavaScript pada *frontend*. Berdasarkan hasil implementasi dan pengujian yang telah dilakukan, dapat disimpulkan bahwa:

1. Keberhasilan Implementasi Sistem
   Sistem berhasil diimplementasikan dengan arsitektur tiga lapis (*three-tier architecture*) yang memisahkan *layer* presentasi, aplikasi, dan data secara efektif. Hal ini menciptakan sistem yang modular, *scalable*, dan mudah dipelihara.
2. Efektivitas RESTful API
   Penerapan RESTful API terbukti efektif dalam menyediakan komunikasi yang terstandarisasi antara frontend dan *backend*. Semua endpoint API (GET /api/kategori, GET /api/menu, GET /api/orders) berhasil diuji dengan *response time* yang optimal (22-31 ms) dan status response 200 OK.
3. Peningkatan Efisiensi Operasional
   Sistem berhasil mengurangi beban kerja staf hingga 30% dibandingkan dengan sistem manual konvensional. Hal ini dicapai melalui otomatisasi proses pemesanan, pengelolaan menu dinamis, dan *tracking* status pesanan secara *real-time*.
4. Pengalaman Pengguna yang Optimal:
   Antarmuka pengguna yang responsif dan intuitif memberikan kemudahan dalam navigasi menu, proses pemesanan, dan transaksi. Fitur keranjang belanja, dashboard admin, dan manajemen menu berfungsi dengan baik sesuai dengan kebutuhan fungsional yang telah ditetapkan.
5. Keamanan dan Stabilitas Sistem
   Implementasi multiple security *layer*s termasuk HTTPS encryption, JWT authentication, input validation, dan CORS configuration berhasil menjamin keamanan sistem dari berbagai ancaman cyber.
6. Integrasi *Database* yang Efisien
   Penggunaan MariaDB dengan skema *database* yang terstruktur (7 tabel utama) berhasil mengelola data menu, pesanan, pengguna, dan transaksi dengan konsistensi dan integritas data yang terjaga.

B. *Saran*

Berdasarkan evaluasi sistem dan keterbatasan yang ditemukan, beberapa saran untuk pengembangan selanjutnya adalah:

1. Integrasi *Payment Gateway Real-time*
   Mengintegrasikan sistem dengan *payment gateway* pihak ketiga seperti Midtrans atau Xendit untuk memproses pembayaran secara otomatis dan real-time, serta implementasi webhook untuk konfirmasi pembayaran.
2. Implementasi Notifikasi Real-time
   Penggunaan teknologi *Web*Socket untuk memberikan notifikasi instan kepada pelanggan mengenai status pesanan (sedang diproses, siap diambil) dan *push notification* untuk meningkatkan *user experience*.
3. Pengembangan Fitur Analitik
   Menambahkan *dashboard analytics* dengan visualisasi data penjualan, laporan menu terlaris, dan analisis tren pelanggan untuk membantu pengambilan keputusan bisnis yang lebih baik.
4. Peningkatan Sistem Testing
   Implementasi pengujian otomatis menggunakan *framework* seperti Jest untuk *backend* dan Cypress untuk *frontend* guna meningkatkan keandalan sistem dan mempercepat proses *development*.
5. Ekspansi ke *Platform Mobile*
   Pengembangan aplikasi *mobile* menggunakan *React Native* atau Flutter untuk menjangkau lebih banyak pengguna dan memberikan pengalaman yang lebih optimal di perangkat *mobile*.

REFERENSI

[1] M. K. Z. Sikumbang, N. Oktaviani, dan R. Diansyah, "Rancang Bangun Sistem Informasi Menu Berbasis *Web*site pada PT. Superfood Cita Insani," *JATI (Jurnal Mahasiswa Teknik Informatika)*, vol. 8, no. 5, Okt. 2024.

[2] T. Asra, S. N. Khasanah, dan E. R. Nainggolan, "Rancang Bangun Sistem Informasi Manajemen Restoran Berbasis *Web* pada Warunk Upnormal," *Reputasi: Jurnal Rekayasa Perangkat Lunak*, vol. 4, no. 2,

Mei 2023.
[3] M. G. Prasetya, D. Heksaputra, Y. Wicaksono, dan A. A. Harahap, "Perancangan Aplikasi Pemesanan Menu pada Kafe Ra Kopiran Berbasis *Web*site Menggunakan Metode Waterfall," *Jurnal Teknologi Sistem Informasi*, vol. 5, no. 2, pp. 173–187, Sep. 2024, doi: 10.35957/jtsi.v5i2.9125.
[4] Fathir Rachman, A., Imam Shalahudin, M., Almas Radifa, F., Studi Teknik Informatika, P., Tinggi Teknologi Informasi NIIT, S., Studi Sistem Informasi, P., Asem Dua No, J., Cipete Selatan, K., Cilandak, K., & Selatan, J. (2023). Implementasi *Web*site Full-Stack Menggunakan Teknologi Next.Js, React, Dan Sanity. GENIEMAS: Jurnal Generasi Teknologi Melayani Masyarakat, 2(Agustus), 19–22.]
[5] IBM. (2025). Apa itu REST API (RESTful API)?. Retrieved from https://www.ibm.com/id-id/think/topics/rest-apis. Diakses 17-06-2025.
[6] Wijayanto, A., Rohmadi, A., & Permana, U. Membangun *Web* API dengan Lumen 5.5. Ardhi Wijayanto, 2018.
[7] Relan, K. (2019). Building REST APIs with Flask: Create Python *Web* Services with MySQL. New Delhi: Apress Media LLC.
[8] https://aws.amazon.com/id/what-is/restful-api/
[9] R. T. Fielding and R. N. Taylor, "Principled Design of the Modern *Web* Architecture," ACM Trans. Internet Technol., vol. 2, no. 2, pp. 115–150, 2002, doi: 10.1145/514183.514185.
[10] M. A. Solahudin, I. Kadek, and D. Nuryana, <RANCANG BANGUN SISTEM INFORMASI STAYCATION BERBASIS *WEB* DENGAN IMPLEMENTASI TEKNOLOGI MERN STACK.
[11] D. Hadi Bachtiar, P. Paniran, and I. M. B. Suksmadana, "Perancangan Back-end Api pada Aplikasi *Mobile* Fruityfit Menggunakan Framework Express JS," J. Tek. Mesin, Ind. Elektro Dan Ilmu Kompute, vol. 2, no. 3, pp. 107–117, 2024, [Online]. Available: https://doi.org/10.61132/mars.v2i3.138
[12] I. K. D. Priyowittesa, D. Sulistyo, dan N. Selviandro, "Perancangan dan Pengimplementasian Sistem *Backend* Aplikasi Cafeasy (Studi Kasus: Kafe daerah Bandung)," *e-Proceeding of Engineering*, vol. 11, no. 4, hlm. 5201, Agu. 2024.
[13] Purwanto, T. Analisa perbandingan kinerja REST API dengan framework Flask, Laravel, dan Express JS. Pijar Pemikiran Scientia, 3(4), 2023.
[14] Mardiansyah, A., Nur Kasah, B., Zamzami H, R. PENGENALAN DASAR HTML DAN CSS: LANGKAH PERTAMA DALAM PENGEMBANGAN *WEB*. Abdi Jurnal Publikasi Vol. 3, No. 3, (165-170), January 2025.
[15] U. K. Siregar, T. A. Sitakar, S. Haramain, Z. N. S. Lubis, U. Nadhirah, dan Y. Yahfizham, "Pengembangan *database* management system menggunakan MySQL," S*AINTEK: Jurnal Sains, Teknologi & Komputer*, vol. 1, no. 1, pp. 8–12, Jan. 2024.
[16] Y. S. Siregar, B. O. Sembiring, E. Rahayu, H. Hasdiana, dan R. Franchitika, "Pemanfaatan Aplikasi MySQL untuk Membantu Siswa SMK Swasta Nur Azizi dalam Pengolahan Data," Jurnal Pengabdian Masyarakat (JAPAMAS), vol. 3, no. 2, pp. 229–240, Des. 2024.
[17] K. V. A. Bucko, B. Krasniqi, dan B. Rexha, "Enhancing JWT authentication and authorization in *web* applications based on user behavior history," Computers, vol. 12, no. 4, 2023