PERANCANGAN DAN IMPLEMENTASI ADMIN PORTAL OKAI SEBAGAI SISTEM MANAJEMEN E-COMMERCE TERINTEGRASI DENGAN DASHBOARD ANALITIK

Fajar Ramadhandi Hidayat[1)], Zakiyah Yasmin Aska Dewi[2)], Achmad Naufal Ferdiansyah[3)]
[1)]Program Studi Sistem Informasi, Universitas Pembangunan Veteran Jawa Timur
E-mail : dhandifajar@gmail.com
[2)]Program Studi Sistem Informasi, Universitas Pembangunan Veteran Jawa Timur
E-mail : zakiyahyasmin1@gmail.com
[3)]Program Studi Sistem Informasi, Universitas Pembangunan Veteran Jawa Timur
E-mail : naufansyah31@gmail.com

ABSTRAK

Pengelolaan operasional *e-commerce* berskala menengah di KAMBI menghadapi permasalahan kompleksitas pada manajemen inventaris multi-gudang, inefisiensi kalkulasi ongkos kirim lintas titik, serta validasi pembayaran dan pembagian komisi mitra yang masih dilakukan secara manual. Untuk memecahkan permasalahan tersebut, penelitian ini mengusulkan pengembangan Admin Portal terpusat menggunakan kerangka kerja Laravel dan React yang diintegrasikan dengan antarmuka pemrograman aplikasi (API) pihak ketiga. Pendekatan metode yang digunakan mencakup restrukturisasi basis data dengan tabel relasi pivot untuk pemisahan data master dan ketersediaan fisik, serta penerapan algoritma *multi-origin checkout* untuk menangani pesanan terpisah. Sistem yang telah dikembangkan secara fungsional mencakup implementasi siklus pesanan berbasis *state machine*, integrasi *Payment Gateway* Xendit untuk validasi pembayaran otomatis, serta penanaman API logistik BiteShip untuk pembuatan nomor resi instan. Selain itu, modul pemasaran didigitalisasi melalui pelacakan sistem afiliasi dan *dropship* yang mengkalkulasi komisi langsung pasca-transaksi. Manfaat dari implementasi sistem ini bagi organisasi adalah terciptanya transparansi data keuangan mitra, peningkatan efisiensi distribusi logistik melalui pelacakan seketika (*real-time*), serta tereliminasinya risiko penjualan barang kosong (*overselling*) pada etalase *web*. Kesimpulan dari penelitian ini menunjukkan bahwa arsitektur sistem informasi yang dibangun berhasil menyinkronkan data transaksional antara *backend* dan *frontend* secara akurat, menciptakan ekosistem *e-commerce* yang tangguh, otomatis, dan terbebas dari anomali fungsional.

Kata kunci: afiliasi, *e-commerce*, logistik, *multi-warehouse*, otomatisasi

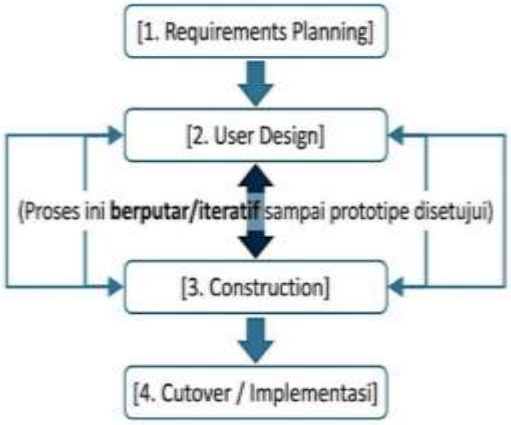
1. PENDAHULUAN

Pertumbuhan pesat dalam industri *e-commerce* menuntut para pelaku usaha untuk memiliki sistem informasi yang adaptif, akurat, dan berorientasi pada kepuasan pelanggan. Pada operasional PT Otak Kanan (KAMBI), kompleksitas pengelolaan data logistik, khususnya pada arsitektur *multi-warehouse* (banyak gudang), seringkali memicu anomali sinkronisasi antara tampilan *dashboard* analitik dengan realita ketersediaan produk di lapangan. Di sisi lain, peningkatan volume transaksi juga berbanding lurus dengan tingginya permintaan layanan pelanggan (*customer service*) untuk menjawab pertanyaan repetitif, seperti pelacakan status resi dan ketersediaan stok barang (An'Nashira, Hamdani and Rahman, 2025). Pendekatan layanan pelanggan konvensional seringkali memakan waktu dan berpotensi menurunkan retensi pengguna. Berdasarkan latar belakang tersebut, rumusan masalah dalam penelitian ini adalah bagaimana merancang dashboard analitik yang mampu menyajikan data transaksi dan stok *multi-warehouse* secara *real-time*, serta bagaimana mengintegrasikan asisten cerdas berbasis *Artificial Intelligence* (AI) yang mampu membaca pangkalan data secara otonom untuk menjawab keluhan pelanggan. Tujuan dari penelitian ini adalah mengimplementasikan arsitektur API dinamis pada backend Laravel untuk menyempurnakan kalkulasi metrik bisnis, serta membangun chatbot asisten menggunakan model Gemini 2.5 Flash yang dilengkapi dengan kemampuan *Function Calling* dan memori jangka panjang (*long-term memory*).

2. LANDASAN TEORI

Metode penelitian yang digunakan dalam pengembangan sistem *e-commerce* ini menggunakan pendekatan *Rapid Application Development* (RAD). RAD merupakan pendekatan siklus hidup pengembangan perangkat lunak yang berfokus pada pengembangan cepat melalui pembuatan prototipe secara iteratif dan umpan balik pengguna yang konstan (Sintawati, 2022), sehingga sangat adaptif terhadap perubahan kebutuhan di lapangan. Artefak pada RAD berpusat pada komponen yang dapat digunakan kembali (*reusable components*) dan prototipe fungsional yang terus disempurnakan berdasarkan evaluasi langsung dari pengguna akhir. Sehingga dapat ditarik sebuah definisi bahwa RAD merupakan kerangka kerja iteratif yang mengutamakan kecepatan, efisiensi waktu, dan kolaborasi antara pengembang dan *stakeholder* untuk menghasilkan perangkat lunak yang fungsional dan berkualitas tinggi (Sarumaha and Junaidi, 2025). Proses ini mencakup empat tahapan utama, yang diaplikasikan pada *platform e-commerce*.

Copyright@2026 FST PSU Bekasi



Gambar 2.1
Tahapan Metode *Rapid Application Development* (RAD)

Requirements Planning (Perencanaan Kebutuhan), adalah tahap pertama yang harus dilakukan dalam metode RAD, yaitu dengan melakukan identifikasi masalah, menentukan batasan sistem, serta mengumpulkan kebutuhan fungsional dan non-fungsional. Pada tahap ini dibutuhkan pemahaman komprehensif dari tim pengembang di PT Otak Kanan untuk berdiskusi dengan pemangku kepentingan guna merumuskan ruang lingkup solusi, seperti kebutuhan integrasi stok *multi-warehouse*, metrik afiliasi, dan fitur *chatbot* asisten AI.

User Design (Desain Pengguna), tahap ini dilakukan dengan membangun interaksi intensif bersama pengguna melalui perancangan antarmuka dan prototipe sistem secara cepat. *Stakeholder* terlibat langsung dalam proses iteratif untuk meninjau dan mengevaluasi rancangan alur kerja, arsitektur basis data (backend Laravel), serta desain antarmuka (*frontend* React) guna memastikan solusi yang dirancang memiliki objektivitas yang tepat terhadap masalah yang dihadapi.

Construction (Konstruksi), pada tahapan ini dilakukan pembuatan dan pengkodean sistem (*coding*) secara aktual berdasarkan prototipe yang telah disetujui pada tahap desain. Fokus utama pada tahap ini adalah mengonversi desain antarmuka menjadi kode fungsional, membangun *Application Programming Interface* (API) yang dinamis, serta mengintegrasikan *Large Language Model* (LLM) Gemini yang dilengkapi dengan Function Calling dan Long-term *Memory* ke dalam *chatbot* bernama Ami. Sistem yang siap beroperasi. Tahapan ini mencakup pengujian fungsionalitas secara menyeluruh, observasi terhadap anomali sistem (seperti evaluasi akurasi perhitungan stok riil), hingga perilisan penuh agar *platform* dan *dashboard* analitik dapat digunakan secara aktual untuk menjawab permasalahan operasional yang telah ditentukan.

3. RANCANGAN SISTEM DAN APLIKASI

3.1 Analisis Kebutuhan Sistem

Kebutuhan fungsional mendeskripsikan seluruh proses, layanan, dan modul yang harus diakomodasi oleh sistem untuk berbagai jenis pengguna (Admin Utama, Admin Gudang, Mitra Afiliasi, dan Pelanggan).

Table 3.1 Kebutuhan Fungsional

Kode	Modul	Kebutuhan Fungsional	Aktor
KF-01	Manajemen Produk	Sistem harus menyediakan fitur untuk menambah, mengubah, dan menghapus (CRUD) data produk, termasuk informasi SKU, harga, kategori, dan gambar.	Admin Utama
KF-02	*Multi Warehouse*	Sistem harus memungkinkan penambahan entitas gudang baru beserta lokasi spesifiknya.	Admin Utama
KF-03	Manajemen Stok	Sistem harus dapat memperbarui (mutasi) stok produk secara spesifik berdasarkan ID Gudang dan ID Produk pada tabel relasi product_warehouses	Admin Gudang
KF-04	Sistem Afiliasi	Sistem harus dapat melacak sumber penjualan (*referral*) dengan menyimpan affiliate_id pada setiap transaksi yang masuk melalui tautan mitra.	Sistem /Pelanggan
KF-05	Manajemen Afiliasi	Sistem harus mampu mengkalkulasi komisi afiliasi secara otomatis berdasarkan persentase dari total pesanan dengan status lunas (*paid*).	ɔistem / Mitra
KF-06	Manajemen Transaksi	Sistem harus menyediakan antarmuka untuk memantau, memvalidasi pembayaran, dan memperbarui status pesanan (*pending*, *paid*, *shipped*, *delivered*).	Admin / Admin Gudang
KF-07	Logistik & Resi	Sistem harus memungkinkan admin untuk memasukkan nomor resi ekspedisi (*waybill*) pada pesanan yang berstatus *shipped*.	Admin Gudang
KF-08	*Dashboard* Analitik	Sistem harus memvisualisasikan data statistik *real-time*, grafik pendapatan bulanan, proporsi kategori produk,	Admin Utama

		dan daftar produk terlaris.
KF-09	Asisten Virtual (Ai)	Sistem harus menyediakan Pelanggan chatbot terintegrasi LLM yang mampu melacak status resi pesanan dan memberikan informasi katalog produk kepada pelanggan.

Cutover (Peralihan atau Implementasi), pada tahap ini hasil dari proses konstruksi (prototipe akhir sistem *e-commerce*) diperlihatkan, diuji, dan dialihkan menjadi Kebutuhan non-fungsional memastikan sistem berjalan dengan aman, cepat, dan memiliki integritas data yang tinggi (Aini *et al.*, 2025), khususnya terkait transaksi keuangan dan stok barang.

Table 3.2 Kebutuhan Non-Fungsional

Kode	**Parameter**	**Kebutuhan Non-Fungsional**
KNF-01	*Security* (Keamanan)	Sistem harus menyediakan fitur untuk menambah, mengubah, dan menghapus (CRUD) data produk, termasuk informasi SKU, harga, kategori, dan gambar.
KNF-02	*Data Integrity* (Integritas)	Kalkulasi data pada *Dashboard* Analitik harus menggunakan teknik *query* agregasi (join dan sum) yang efisien untuk menghindari beban peladen (*server overload*) saat memproses ribuan baris data orders dan order_items
KNF-03	*Performance* (Kinerja)	Penarikan data pesanan pengguna baik melalui sistem *tracking* manual maupun via *Chatbot* wajib mengimplementasikan validasi ID Pengguna untuk mencegah celah *Insecure Direct Object Reference* (IDOR).
KNF-04	*Usability* (Kebergunaan)	Antarmuka pengguna (*frontend*) React harus bersifat responsif, menyajikan navigasi *dashboard* yang intuitif, serta memberikan umpan balik (*feedback*) berupa notifikasi atau indikator pemuatan (*loading*) pada setiap interaksi aksi.

3.2 Perancangan Pengolahan Data



Gambar 3.1
Struktur *Database* pada phpMyAdmin

Untuk mendukung pengelolaan data operasional serta penyajian informasi pada dashboard analitik, sistem dirancang menggunakan basis data relasional yang menyimpan berbagai data *e-commerce*, seperti data produk, pelanggan, transaksi, afiliasi, promosi, hingga aktivitas sistem. Struktur basis data disusun agar mampu menjaga keterkaitan antar entitas serta mendukung proses pengolahan data secara efisien. Divisualisasikan melalui *Entity Relationship Diagram* (ERD) sehingga alur relasi antar tabel dapat dipahami dengan lebih jelas (Ranaputra *et al.*, 2025). Relasi tersebut menjadi dasar dalam proses pengambilan dan pengolahan data, khususnya pada data transaksi, produk, pelanggan, dan afiliasi yang digunakan untuk menghasilkan informasi operasional.



Gambar 3.2
Entity Relationship Diagram (ERD) *Database*

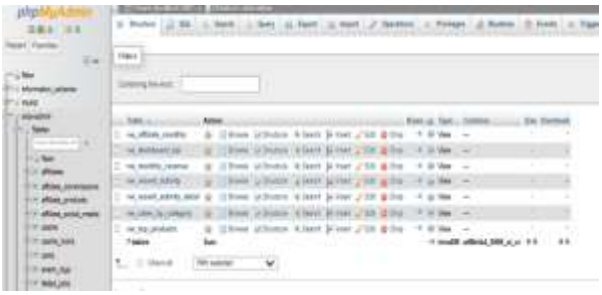
Dalam implementasinya, aplikasi menerapkan arsitektur *client-server* dengan memanfaatkan React.js sebagai *frontend*, Laravel sebagai *backend*, dan MySQL sebagai sistem manajemen basis data. Frontend mengirimkan permintaan (*request*) kepada *backend* melalui REST API, kemudian backend memproses permintaan

tersebut, melakukan interaksi dengan *database*, dan mengembalikan data yang selanjutnya ditampilkan kepada pengguna. Mekanisme ini memungkinkan proses pengelolaan data berlangsung secara terintegrasi sekaligus memisahkan logika bisnis, antarmuka pengguna, dan penyimpanan data.

Hasil pengolahan data kemudian disimpan dalam beberapa SQL *View* yang berfungsi sebagai *data source* bagi *dashboard* analitik. Pendekatan ini dipilih agar proses pengambilan data menjadi lebih efisien, mengurangi pengulangan *query* yang kompleks, serta menjaga konsistensi informasi yang ditampilkan pada aplikasi maupun *dashboard Business Intelligence*. Sebelum digunakan, data melalui proses validasi untuk memastikan kesesuaian struktur, konsistensi relasi antar tabel, serta kelengkapan atribut sehingga informasi yang dihasilkan memiliki tingkat akurasi yang baik.

3.3 Perancangan Pengolahan dan Penyajian Data

Untuk mendukung kebutuhan *monitoring* operasional dan analisis bisnis, sistem dirancang dengan mekanisme pengolahan data yang mengubah data operasional menjadi informasi yang mudah dipahami oleh pengguna. Data yang tersimpan pada basis data diproses menggunakan *query* SQL untuk menghasilkan berbagai metrik yang dibutuhkan dalam proses analisis, seperti total pendapatan, jumlah pelanggan, jumlah produk terjual, serta data aktivitas pengguna.



Gambar 3.3
SQL *View* sebagai *Data Source Dashboard* Analitik

Hasil pengolahan tersebut kemudian disimpan dalam beberapa SQL *View* yang berfungsi sebagai *data source* bagi dashboard analitik. Penggunaan SQL *View* memungkinkan proses pengambilan data menjadi lebih efisien karena logika *query* yang kompleks telah disimpan pada basis data sehingga dapat digunakan kembali oleh aplikasi maupun *dashboard* tanpa perlu melakukan pengolahan berulang. Sebelum digunakan, data juga melalui proses validasi untuk memastikan konsistensi hasil *query* dengan data yang tersimpan pada basis data, sehingga informasi yang dihasilkan memiliki tingkat akurasi yang baik.



Gambar 3.4
Dashboard Analitik OKAI Admin

Dashboard analitik memanfaatkan SQL *View* sebagai sumber data utama untuk menampilkan berbagai *Key Performance Indicator* (KPI) yang mendukung proses monitoring operasional. KPI yang disajikan meliputi Total *Revenue*, *Products Sold*, *Total Customers*, *Average Order Value*, *Revenue Growth*, *Sales by Category*, *Top Products*, serta *Recent Activity*. Penyajian informasi dilakukan melalui berbagai bentuk visualisasi, seperti *card*, grafik batang, grafik garis, dan diagram lingkaran, sehingga pengguna dapat memahami kondisi operasional perusahaan secara lebih cepat dan intuitif.

Melalui mekanisme tersebut, proses pengolahan dan penyajian data mampu menghasilkan informasi yang terstruktur, akurat, dan mudah dipahami sehingga mendukung proses evaluasi kinerja serta pengambilan keputusan berbasis data.

4. HASIL DAN PEMBAHASAN

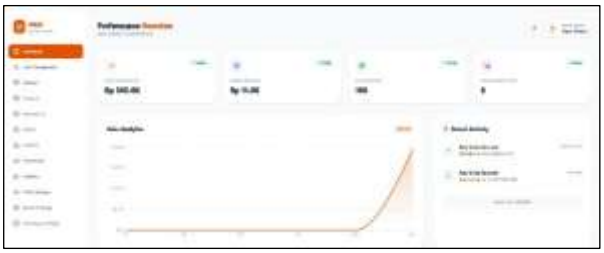
Bagian ini menguraikan hasil implementasi antarmuka dan logika sistem berbasis *framework* Laravel dan React, serta pembahasan mengenai bagaimana sistem menyelesaikan permasalahan bisnis pada operasional *e-commerce* KAMBI.

4.1 Implementasi Arsitektur *Multi-Warehouse* dan *Dashboard* Analitik

Sistem mengimplementasikan arsitektur *multi-warehouse* untuk mendukung pengelolaan stok produk pada lebih dari satu gudang. Pada implementasi ini, data master produk dipisahkan dari data ketersediaan stok fisik dengan memanfaatkan tabel relasi product_warehouses. Pendekatan tersebut memungkinkan setiap gudang memiliki informasi stok yang tersimpan secara terpisah, sementara sistem melakukan proses agregasi untuk memperoleh total stok setiap produk.

Hasil implementasi menunjukkan bahwa mekanisme agregasi data berhasil mengatasi permasalahan pembacaan stok yang sebelumnya hanya bergantung pada tabel products. Dengan memanfaatkan data dari tabel product_warehouses, informasi stok yang ditampilkan menjadi lebih akurat sehingga administrator dapat memantau ketersediaan produk dari berbagai gudang secara *real-time*.

Copyright@2026 FST PSU Bekasi



Gambar 4.1
Dashboard Analitik

Selain mendukung pengelolaan stok, sistem juga mengimplementasikan *dashboard* analitik sebagai media *monitoring* operasional. *Dashboard* menyajikan berbagai informasi penting, seperti total pendapatan (Total *Revenue*), pengiriman aktif (*Active Shipment*), serta grafik pertumbuhan pendapatan bulanan (*Monthly Revenue Growth*). Seluruh data yang ditampilkan diperoleh melalui proses pengolahan pada sisi *backend* menggunakan *query* agregasi dan SQL *View* sebelum dikirimkan ke *frontend* melalui REST API.

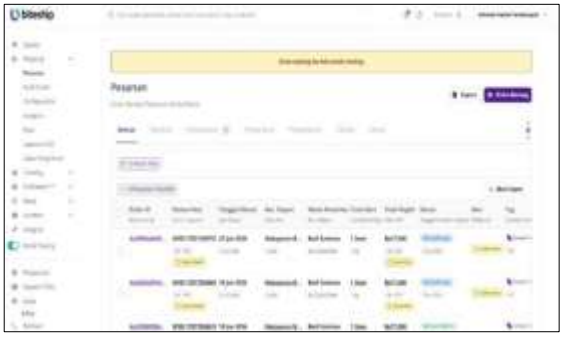


Gambar 4.2
Analytics Management

Hasil implementasi menunjukkan bahwa *dashboard* mampu menyajikan informasi operasional secara terpusat dan mudah dipahami oleh administrator. Penggunaan *query* agregasi pada sisi *backend* juga membantu mengurangi beban komputasi pada sisi klien (*client-side*) sehingga waktu muat (*load time*) halaman menjadi lebih cepat, sekaligus menjaga konsistensi data yang ditampilkan pada *dashboard* analitik.

4.2 Integrasi API Logistik Pihak Ketiga dan Otomatisasi Pembaruan Resi

Kompleksitas operasional *e-commerce* seringkali bermuara pada proses pengiriman dan pelacakan barang (Fadilah and Amirah, 2025) . Untuk mengatasi inefisiensi pada kalkulasi ongkos kirim manual, sistem ini diintegrasikan dengan antarmuka pemrograman aplikasi (API) pihak ketiga, yaitu BiteShip. Pembahasan pada fungsionalitas ini menyoroti kapabilitas sistem dalam mengkalkulasi tarif pengiriman secara dinamis berdasarkan lokasi *warehouse* (gudang) spesifik tempat stok produk dialokasikan menuju titik alamat pelanggan.



Gambar 4.3
Biteship *Dashboard*

Lebih lanjut, integrasi ini memungkinkan otomatisasi pembuatan nomor resi ekspedisi (*waybill generation*). Ketika pesanan diproses, sistem *backend* Laravel melakukan pemanggilan (*HTTP Request*) ke peladen BiteShip dan secara otomatis menyimpan data resi (waybill_id) serta nama kurir ke dalam tabel orders. Sinkronisasi status pengiriman (seperti *shipped* dan *delivered*) yang berjalan secara *real-time* ini tidak hanya memangkas beban kerja admin gudang, tetapi juga menjadi sumber data utama yang diakses oleh asisten virtual (AI) untuk memberikan laporan pelacakan yang akurat kepada pelanggan.

4.3 Pengelolaan Transaksi dan Sistem Afiliasi

Sistem berhasil mengakomodasi pelacakan sumber penjualan melalui penanaman atribut affiliate_id pada setiap entitas pesanan. Pembahasan pada modul ini menitikberatkan pada otomatisasi perhitungan komisi. Ketika admin gudang atau sistem memvalidasi status pesanan menjadi lunas (*paid*) setelah pembayaran dilakukan melalui Integrasi API Payment Gateway Xendit, sistem secara otomatis mengkalkulasi proporsi pendapatan afiliasi tanpa memerlukan intervensi manual. Hal ini meminimalisir kesalahan manusia (*human error*) dalam pembagian hasil dan meningkatkan transparansi bagi mitra afiliasi.



Gambar 4.4
Dashboard Affiliator

Copyright@2026 FST PSU Bekasi

4.4 Pengujian Sistem (*Black Box Testing*)

Pengujian fungsionalitas sistem dilakukan menggunakan metode *Black Box Testing* untuk memvalidasi alur kerja antara *Admin Portal* (sebagai pusat kendali/ *backend*) dengan dampaknya terhadap etalase web *e-commerce* (*frontend*) (Ideal, Rasyid and Yuda, 2024). Skenario pengujian difokuskan pada modul inti seperti manajemen produk, pesanan, afiliasi, dan logistik. Hasil pengujian dirangkum pada Tabel 4.1.

Tabel 4.1. Ringkasan Pengujian Fungsionalitas Inti

No	Modul	Skenario Pengujian	Hasil yang Diharapkan (*Expected Result*)	Hasil Pengujian	Status
1	Manajemen Produk	Admin mengubah nilai is_active menjadi nonaktif (0) pada tabel produk.	Produk tersebut otomatis disembunyikan dari halaman katalog *web e-commerce* pelanggan.	Sesuai harapan	Valid
2	Multi Ware house	Admin Gudang memutasi atau memperbarui nilai stock pada lokasi Gudang tertentu.	Agregasi total stok pada halaman detail produk di web e-commerce menyesuaikan secara real-time.	Sesuai harapan	Valid
3	Manajeme Pesanan	Admin memperbarui status pesanan dari Pending menjadi Paid.	Sistem mengunci alokasi inventaris dan mengubah antarmuka status transaksi di dasbor akun pelanggan.	Sesuai harapan	Valid
4	Logistik	Admin memasukkan nomor resi pada kolom waybill_id di menu Logistics.	Status berubah menjadi Shipped dan pelanggan dapat melacak pengiriman langsung dari web e-commerce.	Sesuai harapan	Valid
5	Sistem Afiliasi	Pelanggan menyelesaikan pesanan (berstatus Paid) melalui tautan referral (affiliate_id aktif).	Sistem mengkalkulasi komisi mitra secara otomatis berdasarkan commission_type (persen/tetap) pada produk terkait.	Sesuai harapan	Valid

Berdasarkan Tabel 4.1, seluruh skenario pengujian yang melibatkan manipulasi data melalui *Admin Portal* terbukti memberikan keluaran yang tepat pada sistem *e-commerce* pelanggan. Hal ini menegaskan bahwa integritas data relasional antar modul seperti kalkulasi diskon *dropship*, pembagian komisi afiliasi, dan sinkronisasi status logistik telah berfungsi secara optimal dan terbebas dari anomali fungsional.

5. KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan hasil perancangan, implementasi, dan pengujian sistem pada *Admin Portal* OKAI / KAMBI E-Commerce, dapat ditarik beberapa kesimpulan utama sebagai berikut:

a. Sentralisasi Kontrol Operasional E-Commerce: Pengembangan *Admin Portal* berhasil mengintegrasikan seluruh modul krusial yang berdampak langsung pada etalase web pelanggan. Pengaturan status keaktifan produk (is_active), harga, manajemen promosi, hingga konfigurasi beranda (*Homepage Settings*) dapat dikontrol secara dinamis tanpa perlu mengubah kode sumber.

b. Integrasi Arsitektur Sistem: Penerapan arsitektur berbasis React.js, Laravel, MySQL, dan REST API berhasil mengintegrasikan antarmuka pengguna, logika bisnis, serta basis data dalam satu sistem yang saling terhubung. Pendekatan ini memungkinkan proses pengelolaan data berlangsung secara konsisten, mendukung komunikasi antar modul, serta meningkatkan efisiensi pengembangan dan pemeliharaan aplikasi.

c. Arsitektur *Multi-Warehouse* dan Algoritma *Multi-Origin Checkout*: Penggunaan tabel relasi pivot (product_warehouses) terbukti sukses mengatasi anomali pembacaan stok dan menghilangkan risiko *overselling*. Lebih lanjut, sistem secara cerdas mampu menangani skenario pesanan kompleks melalui fitur *multi-origin checkout*. Sistem secara otomatis memecah pesanan (*split order*) dan mengakumulasi total ongkos kirim secara presisi jika pelanggan memesan barang dari lokasi gudang fisik yang berbeda.

d. Implementasi *Dashboard* Analitik: *Dashboard* analitik berhasil menyajikan berbagai indikator kinerja utama (*Key Performance Indicator* atau KPI) seperti total pendapatan, jumlah pelanggan, produk terjual, dan pertumbuhan pendapatan dalam bentuk visualisasi interaktif. Penyajian informasi secara terpusat mempermudah administrator dalam melakukan *monitoring* operasional serta mendukung

proses pengambilan keputusan berbasis data secara lebih efektif.

e. Optimalisasi Pengolahan Data: Pemanfaatan SQL *View* sebagai sumber data *dashboard* mampu menyederhanakan proses pengambilan informasi dari basis data. Pendekatan ini mampu menyederhanakan proses pengambilan data, mengurangi kompleksitas *query* pada aplikasi, menjaga konsistensi informasi, serta meningkatkan efisiensi penyajian data pada *dashboard* analitik.
f. Otomatisasi Finansial dan Logistik Terintegrasi: Integrasi *Payment Gateway* Xendit dan API Logistik (BiteShip) berhasil mewujudkan otomatisasi siklus pesanan (*Order Lifecycle*). Melalui sistem *Instant Callback*, perubahan status transaksi dari *Pending* menjadi *Paid* terjadi seketika, diikuti dengan pembuatan nomor resi otomatis (waybill_id) yang menyinkronkan status pengiriman ke akun pelanggan.
g. Transparansi Sistem Afiliasi dan *Dropship*: Modul pemasaran kolaboratif berhasil didigitalisasi dengan tingkat akurasi tinggi. Validasi transaksi yang otomatis terpicu oleh sistem pembayaran Xendit langsung mengkalkulasi saldo komisi mitra berdasarkan data rujukan (affiliate_id), serta menerapkan pemotongan harga otomatis bagi akun *dropshipper*.

5.2 Saran

Meskipun sistem telah berjalan dengan optimal dan memenuhi spesifikasi kebutuhan bisnis PT Otak Kanan, beberapa peluang pengembangan untuk penelitian selanjutnya adalah sebagai berikut:

a. Penerapan Prediksi Inventaris (*Forecasting*) Berbasis Kecerdasan Buatan: Mengingat sistem telah memiliki modul *multi-warehouse* yang solid, penelitian selanjutnya dapat menambahkan algoritma prediksi analitik (*predictive analytics*). Algoritma ini dapat menganalisis data riwayat penjualan bulanan untuk memberikan rekomendasi kepada admin terkait alokasi restok barang di gudang-gudang tertentu sebelum stok tersebut habis.
b. Pengembangan *Dashboard* Analitik: *Dashboard* analitik dapat dikembangkan dengan menambahkan fitur visualisasi interaktif, filter data yang lebih fleksibel, serta analisis prediktif berbasis machine learning sehingga sistem tidak hanya menyajikan data historis, tetapi juga mampu memberikan rekomendasi dalam mendukung pengambilan keputusan.
c. Fitur *Automated Refund* via *Payment Gateway*: Untuk menyempurnakan otomatisasi finansial, sistem dapat ditambahkan fitur pengembalian dana (*refund*) otomatis melalui API Xendit jika terjadi pembatalan transaksi secara sepihak atau apabila terjadi kendala operasional sebelum barang dikirim oleh pihak logistik.
d. Aplikasi *Mobile* untuk *Warehouse Management System* (WMS): Penelitian selanjutnya dapat mengembangkan aplikasi *mobile* WMS yang terintegrasi langsung dengan API Admin Portal. Aplikasi tersebut dapat memanfaatkan fitur pemindai kode batang (*barcode scanner*) pada perangkat seluler untuk mendukung proses penyiapan (*picking*) dan pengemasan (*packing*) pesanan. Implementasi aplikasi *mobile* ini diharapkan mampu meningkatkan efisiensi operasional gudang sekaligus memperkuat integrasi proses logistik dalam ekosistem *e-commerce*.

DAFTAR PUSTAKA

Ideal, M.A.V., Rasyid, M. and Yuda, F. (2024) 'Perancangan Sistem Informasi Monitoring Praktek Kerja Lapangan dengan Menggunakan Metode Waterfall,' Jurnal KomtekInfo, pp. 425–435. https://doi.org/10.35134/komtekinfo.v11i4.595.

Sintawati, I.D. (2022) 'KOMPARASI METODE RAD DENGAN RUP PADA PENGEMBANGAN SISTEM INFORMASI,' Akrab Juara Jurnal Ilmu-ilmu Sosial, 7(2), p. 101. https://doi.org/10.58487/akrabjuara.v7i2.1796.

Sarumaha, D. and Junaidi (2025) 'Perancangan Aplikasi e-Lades sebagai Sistem Pelayanan Administrasi Masyarakat Berbasis Website di Desa XYZ menggunakan Metode Rapid Application Development,' JURIKOM (Jurnal Riset Komputer), 12(6), pp. 827–836. https://doi.org/10.30865/jurikom.v12i6.9341.

Aini, S.N. et al. (2025) 'Perancangan sistem absensi pada tingkat perguruan tinggi negeri berbasis QR code,' Jurnal Rekayasa Sistem Informasi Dan Teknologi, 2(4), pp. 1457–1467. https://doi.org/10.70248/jrsit.v2i4.2412.

Ranaputra, A. et al. (2025) 'Implementasi E-Commerce berbasis web pada SAponik Farm,' Jurnal Komputer Antartika, 3(4), pp. 145–154. https://doi.org/10.70052/jka.v3i4.1210.

Fadilah, K.N. and Amirah, A. (2025) 'Manajemen Risiko Operasional pada Prusahaan Ekspedisi Pengiriman di Kabupaten Brebes,' Jurnal Nuansa Publikasi Ilmu Manajemen Dan Ekonomi Syariah, 3(4), pp. 249–263. https://doi.org/10.61132/nuansa.v3i4.2430.

An'Nashira, S., Hamdani, A. and Rahman, A.Z. (2025) 'Analisis Kualitas Pelayanan Pengiriman Barang Terhadap Pelanggan Ekspedisi J&T,' Jurnal Sosial Dan Sains, 5(4), pp. 1048–1057. https://doi.org/10.59188/jurnalsosains.v5i4.32157.