TRANSCIVILIA: Journal of Multidisciplinary Knowledge and Civilization

Research Paper **ISSN:**

Desain dan Implementasi Sistem Informasi *E-Commerce Multi-Tenant* Berbasis Next.js dengan Integrasi *Payment Gateway* Menggunakan Metode *Prototype*

[1]Adhim Awalul Khakim, [2]Ahnaf Nafi, [3]Rizki Prasetyo Tulodo

[1,3]Department of Informatics, Pancasakti Tegal University, Indonesia

[2]Department of Islamic Studies, Walisongo State Islamic University Semarang, Indonesia

Email: adhimawalul2005@gmail.com[1], 23030360069@student.walisongo.ac.id[2], 31169101992@upstegal.ac.id[3]

Abstract:

The rapid growth of e-commerce in Indonesia has increased the need for marketplace platforms that can efficiently manage multiple tenants within a single ecosystem. This study aims to design and implement a Next.js-based multi-tenant e-commerce information system with an integrated payment gateway to support administrators, tenants, and customers. The system was developed using the Prototype method through iterative design, development, and user evaluation. The proposed system integrates role-based dashboards, centralized data management, and the Midtrans payment gateway with automated payment status updates and notifications. System validation using black-box testing showed that all core functions operated according to the specified requirements. User evaluation further indicated that the system effectively supported store management, transaction processing, and online payment activities with satisfactory performance. This study contributes a practical implementation model for developing a Next.js-based multi-tenant e-commerce platform that integrates user management, transaction processing, payment services, and notification mechanisms within a unified architecture. The proposed model can serve as a reference for developing independent marketplace platforms, particularly for educational institutions and MSME communities.

Keywords: *E-Commerce, Multi-Tenant, Next.js, Midtrans, Prototype.*

TRANSCIVILIA: Journal of Multidisciplinary Knowledge and Civilization
Vol 01 Issue 04 2026
Received: 27/06/2026 Revised: 29/06/2026 Accepted: 03/08/2026 Online: 03/08/2026

Introduction

Transformasi ekonomi digital di Indonesia dalam beberapa tahun terakhir berlangsung semakin pesat seiring meningkatnya penetrasi internet dan adopsi teknologi informasi pada berbagai sektor kehidupan. Perkembangan tersebut tidak hanya mengubah pola komunikasi masyarakat, tetapi juga mendorong perubahan perilaku konsumsi dari transaksi konvensional menuju transaksi digital. Laporan Asosiasi Penyelenggara Jasa Internet Indonesia (APJII) menunjukkan bahwa jumlah pengguna internet di Indonesia telah mencapai lebih dari 221 juta jiwa, yang mencerminkan semakin luasnya peluang pemanfaatan platform digital sebagai sarana aktivitas ekonomi dan perdagangan (Prasetiyo et al., 2024). Kondisi ini menjadikan e-commerce sebagai salah satu sektor yang mengalami pertumbuhan paling cepat sekaligus mendorong kebutuhan akan platform yang mampu melayani banyak pelaku usaha secara efisien.

Salah satu pendekatan yang banyak diterapkan untuk memenuhi kebutuhan tersebut adalah model *multi-tenant*, yaitu arsitektur yang memungkinkan banyak tenant beroperasi pada satu infrastruktur aplikasi yang sama dengan tetap mempertahankan pengelolaan data dan layanan secara terpisah. Dibandingkan dengan pengembangan toko daring secara mandiri, pendekatan ini menawarkan efisiensi biaya pengembangan, pemeliharaan, dan operasional karena seluruh tenant memanfaatkan sumber daya sistem yang sama (Pranoto, 2026). Keberhasilan platform marketplace seperti Tokopedia, Shopee, dan Lazada menunjukkan bahwa model bisnis multi-tenant mampu

mengakomodasi aktivitas perdagangan digital dalam skala besar sekaligus mempertemukan penjual dan pembeli dalam satu ekosistem yang terintegrasi (Satria et al., 2023).

Meskipun demikian, karakteristik marketplace komersial tersebut belum sepenuhnya dapat memenuhi kebutuhan institusi pendidikan, koperasi, maupun komunitas UMKM yang memerlukan kendali penuh terhadap pengelolaan data, kebijakan layanan, dan manajemen pengguna (Sammir & Hamdi, 2025). Pada kondisi tersebut, pengembangan sistem e-commerce secara mandiri menjadi alternatif yang lebih strategis karena memberikan fleksibilitas dalam menyesuaikan kebutuhan organisasi. Konsekuensinya, sistem yang dikembangkan tidak hanya dituntut mampu mendukung proses transaksi, tetapi juga harus menyediakan mekanisme pengelolaan admin, tenant, dan customer dalam satu ekosistem yang aman, mudah dikembangkan, serta memiliki performa yang baik (Namira Syahputri, 2024). Kebutuhan tersebut mendorong pemanfaatan teknologi web modern yang mampu mengintegrasikan pengelolaan antarmuka, logika bisnis, akses basis data, hingga layanan pembayaran secara efisien.

Berbagai penelitian mengenai pengembangan sistem e-commerce telah banyak dilakukan, baik berbasis web maupun perangkat bergerak, untuk membantu pelaku usaha memasarkan produknya secara digital (Perillo, 2024). Namun, sebagian besar penelitian tersebut masih memanfaatkan framework PHP konvensional, seperti CodeIgniter dan Laravel, dengan arsitektur monolitik (Abd-elfatah Tamimi, Thamer AL-Rawashdeh, 2021). Walaupun framework tersebut telah terbukti stabil dan banyak digunakan, perkembangan teknologi web modern menghadirkan alternatif yang menawarkan peningkatan performa dan pengalaman pengguna. Next.js, melalui dukungan Server-Side Rendering (SSR), Static Site Generation (SSG), serta App Router, memungkinkan proses pemuatan halaman menjadi lebih cepat, meningkatkan optimasi mesin pencari (*Search Engine Optimization*), serta menghasilkan antarmuka yang lebih responsif (Lipiński & Pańczyk, 2023). Keunggulan tersebut menjadikan Next.js relevan untuk diterapkan pada pengembangan sistem e-commerce yang membutuhkan skalabilitas dan performa tinggi.

Meskipun demikian, kajian mengenai implementasi Next.js pada sistem e-commerce multi-tenant masih relatif terbatas, khususnya dalam konteks penelitian di Indonesia (Samuel Nugraha, 2023). Penelitian terdahulu cenderung membahas aspek-aspek tersebut secara terpisah, seperti implementasi framework web, pengembangan marketplace, atau integrasi layanan pembayaran, tanpa menggabungkannya ke dalam satu arsitektur yang utuh (Harshit Mishra, Harshit Shukla, 2024). Belum banyak penelitian yang mengintegrasikan arsitektur Next.js App Router, manajemen tiga peran pengguna (admin, tenant, dan *customer*), mekanisme *webhook payment gateway*, serta sistem notifikasi secara real-time ke dalam satu arsitektur e-commerce multi-tenant yang terpadu. Kesenjangan tersebut menjadi dasar penelitian ini sekaligus menunjukkan adanya peluang untuk mengembangkan model implementasi yang lebih komprehensif dibandingkan penelitian-penelitian sebelumnya.

Berdasarkan uraian tersebut, penelitian ini bertujuan mengembangkan sistem informasi e-commerce multi-tenant berbasis Next.js menggunakan metode Prototype dengan mengintegrasikan delapan entitas basis data, dashboard terpusat untuk tiga peran pengguna, serta payment gateway Midtrans melalui mekanisme webhook. Kontribusi penelitian ini tidak hanya berupa implementasi sebuah prototipe sistem, tetapi juga menawarkan model arsitektur e-commerce multi-tenant yang mengintegrasikan teknologi web modern, pengelolaan multi-peran, dan otomasi transaksi pembayaran dalam satu rancangan yang terpadu (Ari Yogi Prasetyo, 2024). Model tersebut diharapkan dapat menjadi referensi bagi pengembangan marketplace mandiri pada institusi pendidikan, koperasi, maupun komunitas UMKM yang memerlukan sistem dengan tingkat fleksibilitas, skalabilitas, dan kendali pengelolaan yang lebih tinggi.

Literature Review

Sistem informasi merupakan sekumpulan komponen yang saling berinteraksi untuk mengumpulkan, mengolah, menyimpan, serta mendistribusikan informasi guna mendukung proses pengambilan keputusan dalam suatu organisasi (Mehmannavaz, 2013). Seiring berkembangnya transformasi digital, sistem informasi tidak lagi berfungsi sebagai media pengelolaan data, tetapi juga menjadi fondasi dalam mengintegrasikan berbagai proses bisnis secara otomatis guna meningkatkan efisiensi operasional dan kualitas layanan. Salah satu implementasinya adalah *electronic commerce (e-commerce)*, yaitu aktivitas perdagangan yang memanfaatkan media elektronik untuk mendukung proses pemesanan, pembayaran, hingga distribusi produk atau layanan (Kenneth C. Laudon, 2014). Integrasi sistem informasi dengan e-commerce memungkinkan otomatisasi proses bisnis sehingga transaksi menjadi lebih cepat, akurat, dan transparan. Hal tersebut menunjukkan bahwa efektivitas sebuah platform e-commerce tidak hanya ditentukan oleh kemampuan memproses transaksi, tetapi juga oleh kemampuannya mengintegrasikan data, pengguna, dan layanan dalam satu sistem yang terpadu. Oleh karena itu, penelitian ini memanfaatkan konsep sistem informasi sebagai landasan dalam merancang platform e-commerce multi-tenant yang mampu mengelola berbagai proses bisnis secara terintegrasi.

Model *multi-tenant marketplace* merupakan salah satu bentuk implementasi e-commerce yang memungkinkan beberapa tenant menjalankan aktivitas bisnis pada satu platform yang sama dengan tetap

mempertahankan pemisahan data dan pengelolaan layanan masing-masing (Jadhav, Bandgar, Fattepurkar, 2014). Dalam model ini, administrator bertanggung jawab mengelola ekosistem secara menyeluruh, mulai dari verifikasi tenant, manajemen pengguna, pemantauan transaksi, hingga pengelolaan notifikasi sistem. Sementara itu, tenant berfokus pada pengelolaan toko dan produknya, sedangkan customer berinteraksi melalui proses pencarian produk, transaksi, hingga pembayaran. Pembagian peran tersebut menjadikan model multi-tenant lebih efisien dalam mengelola banyak pelaku usaha pada satu platform, namun di sisi lain meningkatkan kompleksitas sistem dibandingkan e-commerce konvensional. Tantangan utama yang dihadapi meliputi isolasi data antar-tenant, keamanan transaksi, transparansi proses bisnis, serta keandalan mekanisme notifikasi (Jain, 2024). Dengan demikian, keberhasilan implementasi marketplace multi-tenant tidak hanya bergantung pada kemampuan sistem mengakomodasi banyak tenant, tetapi juga pada rancangan arsitektur yang mampu menjaga keamanan, konsistensi data, dan efisiensi pengelolaan platform secara bersamaan. Pertimbangan tersebut menjadi dasar dalam penelitian ini untuk merancang sistem e-commerce multi-tenant yang mendukung pengelolaan berbagai peran pengguna secara terintegrasi sebelum diimplementasikan menggunakan framework Next.js.

Perkembangan teknologi web modern menghadirkan berbagai framework yang dirancang untuk meningkatkan performa aplikasi sekaligus mempermudah proses pengembangan (Vyas, 2022). Salah satu framework yang berkembang pesat adalah Next.js, yaitu framework berbasis React yang dikembangkan oleh Vercel dengan dukungan Server-Side Rendering (SSR), Static Site Generation (SSG), dan Incremental Static Regeneration (ISR). Melalui paradigma App Router yang memanfaatkan React Server Components, proses rendering dapat dilakukan secara lebih efisien sehingga mengurangi beban JavaScript pada sisi klien dan meningkatkan kecepatan pemuatan halaman. Selain itu, fitur API Routes memungkinkan pengembang mengintegrasikan fungsi frontend dan backend dalam satu repositori aplikasi, sehingga proses pengembangan, pemeliharaan, dan deployment menjadi lebih sederhana dibandingkan pendekatan yang memisahkan kedua komponen tersebut (Tadi, 2024). Karakteristik tersebut memberikan keuntungan bagi aplikasi marketplace multi-tenant yang harus melayani banyak pengguna dan transaksi secara bersamaan, karena proses rendering di sisi server mampu mempercepat akses awal halaman, sedangkan integrasi frontend dan backend mendukung pengelolaan sistem yang lebih efisien dalam satu arsitektur. Dengan demikian, Next.js tidak hanya dipilih karena menyediakan fitur-fitur modern, tetapi juga karena mampu memenuhi kebutuhan performa, skalabilitas, dan kemudahan pengembangan yang menjadi karakteristik utama sistem e-commerce multi-tenant. Pemilihan framework ini selanjutnya menjadi dasar dalam mengintegrasikan layanan pembayaran sebagai bagian dari proses transaksi pada sistem yang dikembangkan.

Dalam sistem *e-commerce*, proses pembayaran merupakan salah satu komponen yang menentukan keberhasilan transaksi. Oleh karena itu, keberadaan *payment gateway* berperan penting sebagai penghubung antara pembeli, penjual, dan institusi keuangan dalam memproses transaksi elektronik secara aman (Revi Angeli Siahaan, 2024). Penelitian ini memanfaatkan layanan Midtrans yang menyediakan Snap API sebagai antarmuka pembayaran dan Core API untuk pengelolaan notifikasi transaksi melalui mekanisme webhook. Mekanisme tersebut memungkinkan sistem memperbarui status pembayaran secara otomatis setelah transaksi diproses oleh penyedia layanan pembayaran, sehingga mengurangi ketergantungan pada proses konfirmasi manual dan meminimalkan potensi inkonsistensi data. Selain meningkatkan efisiensi operasional, pendekatan ini juga mendukung sinkronisasi informasi transaksi secara real-time, yang menjadi kebutuhan penting pada marketplace multi-tenant dengan banyak tenant dan transaksi yang berlangsung secara bersamaan. Keamanan komunikasi antarsistem dijamin melalui mekanisme verifikasi tanda tangan digital berbasis SHA-512 sehingga integritas data transaksi tetap terpelihara. Dengan karakteristik tersebut, integrasi Midtrans tidak hanya berfungsi sebagai fasilitas pembayaran, tetapi juga menjadi bagian dari mekanisme pengelolaan transaksi yang mendukung keandalan dan konsistensi sistem *e-commerce* yang dikembangkan (Abd-elfatah Tamimi, Thamer AL-Rawashdeh, 2021).

Keberhasilan implementasi sistem multi-tenant tidak hanya ditentukan oleh aspek teknis, tetapi juga oleh kualitas interaksi pengguna dan sistem. Salah satu komponen yang mendukung hal tersebut adalah *role-based dashboard*, yaitu antarmuka yang mengintegrasikan informasi dan fungsi utama sesuai kebutuhan masing-masing pengguna (Vázquez-ingelmo et al., 2019). Dalam penelitian ini, dashboard dibedakan menjadi tiga jenis berdasarkan peran pengguna. Dashboard administrator berfungsi mengelola keseluruhan ekosistem *platform*, *dashboard tenant* difokuskan pada pengelolaan toko dan transaksi masing-masing, sedangkan dashboard customer menyediakan informasi terkait aktivitas pembelian dan status pembayaran. Pemisahan dashboard berdasarkan peran merupakan implementasi prinsip *usability*, karena setiap pengguna hanya berinteraksi dengan informasi dan fungsi yang relevan terhadap tugasnya sehingga penggunaan sistem menjadi lebih efektif dan intuitif (Aini et al., 2020). Selain meningkatkan kemudahan penggunaan, pendekatan ini juga mendukung penerapan kontrol akses berbasis peran (*role-based access control*) dengan membatasi akses pengguna hanya pada fitur yang sesuai dengan tanggung jawabnya.

Dengan demikian, dashboard terpusat tidak hanya meningkatkan pengalaman pengguna, tetapi juga mendukung efisiensi pengelolaan dan keamanan operasional pada sistem e-commerce *multi-tenant* yang dikembangkan

Pengembangan sistem pada penelitian ini menggunakan metode *Prototype*, yaitu pendekatan pengembangan perangkat lunak yang menekankan pembangunan model awal sistem untuk memperoleh umpan balik pengguna secara cepat sebelum dikembangkan menjadi produk akhir. Menurut Roger Pressman and Bruce Maxim (2020) metode Prototype terdiri atas tiga tahapan utama, yaitu *Listen to Customer, Build/Revise Mock-up, dan Customer Test Drives Mock-up*. Siklus yang bersifat iteratif memungkinkan kebutuhan pengguna divalidasi secara bertahap sehingga risiko ketidaksesuaian antara kebutuhan pengguna dan implementasi sistem dapat diminimalkan. Karakteristik tersebut menjadi relevan untuk pengembangan sistem e-commerce multi-tenant karena melibatkan berbagai aktor, yaitu administrator, tenant, dan customer, yang memiliki kebutuhan fungsional dan alur interaksi yang berbeda. Melalui proses evaluasi pada setiap iterasi, kebutuhan setiap pengguna dapat diakomodasi sebelum sistem diimplementasikan secara penuh, sehingga kualitas antarmuka, proses bisnis, dan integrasi antarfitur dapat disempurnakan secara berkelanjutan (Ari Yogi Prasetyo, 2024). Dengan demikian, metode Prototype dipilih tidak hanya sebagai pendekatan pengembangan perangkat lunak, tetapi juga sebagai strategi untuk memastikan bahwa sistem yang dikembangkan sesuai dengan kebutuhan pengguna dan karakteristik *marketplace multi-tenant*.

Berdasarkan uraian pada enam konsep utama tersebut, dapat disimpulkan bahwa setiap landasan teori saling melengkapi dalam mendukung pengembangan sistem yang diusulkan. Sintesis hubungan antara konsep-konsep tersebut dan relevansinya terhadap penelitian disajikan pada tabel berikut:

Table 1. Sintesis Konsep Utama dan Kontribusinya terhadap Penelitian

Konsep	Kontribusi terhadap Penelitian
Sistem Informasi dan E-Commerce	Menjadi landasan integrasi proses bisnis, pengelolaan data, transaksi, dan layanan dalam satu platform.
Marketplace Multi-Tenant	Menentukan model arsitektur yang mampu mengelola banyak tenant dengan tetap menjaga isolasi data dan keamanan transaksi.
Framework Next.js	Mendukung performa, skalabilitas, dan efisiensi pengembangan melalui SSR, App Router, dan API Routes.
Payment Gateway (Midtrans)	Mendukung otomatisasi pembayaran, pembaruan status transaksi secara *real-time*, dan keamanan komunikasi data.
Dashboard Berbasis Peran	Meningkatkan usability sekaligus mengatur hak akses sesuai peran administrator, tenant, dan customer.
Metode Prototype	Memungkinkan pengembangan sistem secara iteratif sesuai kebutuhan pengguna yang dinamis.

Berdasarkan sintesis pada tabel tersebut, dapat dipahami bahwa setiap konsep memberikan kontribusi yang saling melengkapi terhadap pengembangan sistem e-commerce multi-tenant. Selanjutnya, penelitian terdahulu dianalisis untuk mengidentifikasi perkembangan penelitian yang telah dilakukan serta menemukan celah penelitian yang menjadi dasar pengembangan penelitian ini.

Berbagai penelitian terdahulu menunjukkan bahwa pengembangan sistem *e-commerce* telah dilakukan dari berbagai perspektif, mulai dari integrasi layanan pembayaran, pemanfaatan framework modern, pengelolaan antarmuka pengguna, hingga optimasi basis data. Maulana dkk. (2022) menitikberatkan pada integrasi *payment gateway Midtrans* sehingga konfirmasi pembayaran dapat dilakukan secara otomatis melalui mekanisme webhook. Pada sisi lain, Hanafi dkk. (2024) menunjukkan bahwa Next.js dengan pendekatan *Server-Side Rendering* (SSR) mampu meningkatkan performa aplikasi web, meskipun belum diterapkan pada sistem e-commerce. Pada aspek pengembangan sistem. Sementara itu, Kusnadi dan Putra (2024) membuktikan bahwa framework CodeIgniter efektif digunakan dalam pengembangan e-commerce UMKM, namun masih mengadopsi arsitektur konvensional. Selain itu, penelitian Ismail dkk. (2026) memperlihatkan bahwa dashboard terintegrasi mampu meningkatkan efektivitas pengelolaan operasional, sedangkan Abdillah dkk. (2025) menyoroti adanya trade-off antara kemudahan pengembangan menggunakan ORM dan performa akses basis data. Secara keseluruhan, penelitian-penelitian tersebut menunjukkan bahwa setiap studi cenderung berfokus pada aspek tertentu dari pengembangan sistem e-commerce, sehingga belum menghadirkan integrasi antara arsitektur multi-tenant, framework modern, layanan pembayaran, dan pengelolaan pengguna dalam satu sistem yang terpadu. Temuan tersebut menjadi dasar untuk mengidentifikasi celah penelitian yang dibahas pada bagian berikutnya.

Secara keseluruhan, penelitian-penelitian terdahulu menunjukkan bahwa pengembangan sistem *e-commerce* telah mengalami perkembangan yang signifikan pada berbagai aspek, seperti pemanfaatan framework modern, integrasi *payment gateway*, pengembangan dashboard berbasis peran, serta optimalisasi pengelolaan basis data (Rohunen et al., 2014). Meskipun demikian, kajian-kajian tersebut umumnya masih berfokus pada pengembangan masing-masing komponen secara terpisah sehingga belum menghasilkan suatu arsitektur yang mengintegrasikan seluruh aspek tersebut dalam satu sistem yang utuh. Berdasarkan hasil sintesis literatur, belum ditemukan penelitian yang menggabungkan arsitektur multi-tenant berbasis Next.js App Router, dashboard berbasis tiga peran, integrasi Midtrans melalui mekanisme webhook, serta rancangan basis data yang saling terhubung dalam satu sistem e-commerce (Lipiński & Pańczyk, 2023). Oleh karena itu, penelitian ini mengusulkan pengembangan sistem e-commerce multi-tenant yang mengintegrasikan seluruh komponen tersebut melalui metode Prototype sebagai upaya untuk menghasilkan arsitektur yang lebih terpadu, efisien, dan sesuai dengan kebutuhan pengelolaan marketplace modern. Kontribusi utama penelitian ini terletak pada penyusunan model implementasi yang tidak hanya mengadopsi teknologi-teknologi yang telah ada, tetapi juga mengintegrasikannya menjadi satu kesatuan sistem yang dapat menjadi acuan bagi pengembangan platform marketplace multi-tenant pada penelitian selanjutnya.

Research Method

Penelitian ini merupakan penelitian rekayasa perangkat lunak (*software engineering research*) dengan pendekatan kualitatif-deskriptif yang berorientasi pada perancangan, implementasi, dan validasi sistem informasi e-commerce multi-tenant (Kabbedijk, 2014). Metode pengembangan yang digunakan adalah *Prototype* karena memiliki karakteristik iteratif yang memungkinkan proses penyempurnaan sistem dilakukan secara bertahap berdasarkan umpan balik pengguna. Pendekatan ini dipilih mengingat sistem yang dikembangkan melibatkan tiga kelompok pengguna dengan kebutuhan dan alur kerja yang berbeda, yaitu administrator, tenant, dan customer, sehingga validasi pada setiap iterasi menjadi aspek penting dalam menghasilkan sistem yang sesuai dengan kebutuhan pengguna.

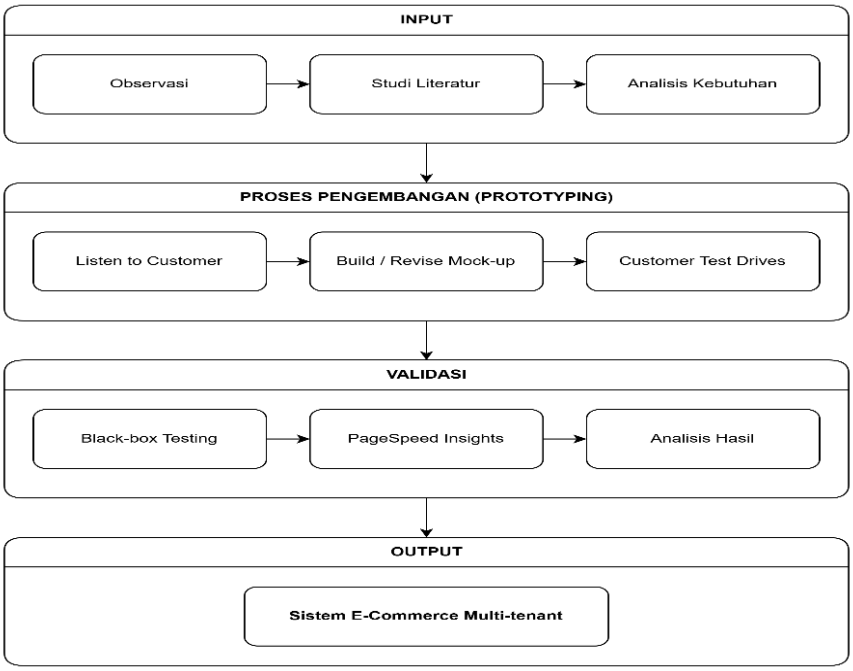

Figure 1. Alur Penelitian (*Flowchart*)

Pengembangan sistem dilaksanakan melalui tiga tahapan utama sesuai siklus *Prototype*, yaitu *Listen to Customer*, *Build* atau *Revise Mock-up*, dan *Customer Test Drives Mock-up*. Tahap *Listen to Customer* bertujuan mengidentifikasi kebutuhan sistem melalui observasi terhadap platform marketplace yang telah beroperasi serta studi literatur dari jurnal ilmiah dan dokumentasi teknis (Maggenta et al., 2022). Hasil identifikasi menghasilkan dua kelompok kebutuhan sistem, yaitu kebutuhan fungsional dan kebutuhan nonfungsional. Kebutuhan fungsional meliputi mekanisme registrasi dan autentikasi multi-peran (administrator, tenant, dan *customer*), pengelolaan toko serta profil tenant oleh administrator, manajemen katalog produk oleh tenant, keranjang belanja persisten, proses checkout terintegrasi, pembayaran menggunakan Midtrans melalui mekanisme webhook, serta notifikasi status pesanan dan

pembayaran secara real-time. Adapun kebutuhan nonfungsional meliputi waktu muat halaman kurang dari dua detik, penggunaan protokol HTTPS pada seluruh rute aplikasi, serta kompatibilitas terhadap peramban modern (Falih, 2019). Tahap ini menjadi dasar penyusunan rancangan sistem sekaligus acuan dalam proses implementasi pada iterasi berikutnya.

Tahap *Build* atau *Revise Mock-up* merupakan proses pembangunan sekaligus penyempurnaan prototipe berdasarkan kebutuhan yang telah diidentifikasi sebelumnya. Sistem dikembangkan menggunakan Next.js 16 dengan paradigma App Router yang mengintegrasikan antarmuka pengguna dan API Routes dalam satu repositori aplikasi (Baehaqi et al., 2023). Pengelolaan basis data dilakukan menggunakan Prisma ORM yang berfungsi sebagai lapisan abstraksi terhadap PostgreSQL, dengan rancangan delapan entitas utama yang terdiri atas *user*, tenant, produk, keranjang, pesanan, detail pesanan, pembayaran, dan notifikasi, beserta relasi antarentitasnya. Proses pembayaran diimplementasikan melalui layanan Midtrans, memanfaatkan Snap API sebagai antarmuka pembayaran dan endpoint webhook /api/payment/notification untuk memperbarui status transaksi secara otomatis. Seluruh antarmuka dikembangkan menggunakan Tailwind CSS guna menghasilkan tampilan yang responsif pada berbagai ukuran perangkat.

Tahap *Customer Test Drives Mock-up* dilakukan untuk mengevaluasi kesesuaian prototipe terhadap kebutuhan pengguna. Evaluasi melibatkan lima responden yang merepresentasikan pengguna akhir (*end-user*) dan pemangku kepentingan utama sistem, yaitu dua hingga tiga pengguna utama (*primary user*) yang mewakili peran administrator, tenant, dan customer, satu pengguna sekunder (*secondary user*), serta satu *product owner* yang merepresentasikan kebutuhan bisnis. Masing-masing responden menjalankan skenario penggunaan sesuai perannya, meliputi verifikasi tenant dan monitoring transaksi oleh administrator, pengelolaan produk oleh tenant, serta proses pembelian mulai dari penelusuran produk hingga konfirmasi pembayaran oleh customer. Umpan balik yang diperoleh pada tahap ini digunakan sebagai dasar penyempurnaan sistem pada iterasi berikutnya sehingga proses pengembangan tetap selaras dengan kebutuhan pengguna (Samuel Nugraha, 2023).

Selain evaluasi pengguna, validasi teknis dilakukan menggunakan black-box testing untuk memverifikasi kesesuaian fungsi sistem terhadap spesifikasi yang telah ditetapkan tanpa mengevaluasi struktur kode atau implementasi internal (Firdhayanti et al., 2023). Pendekatan ini dipilih karena mampu memastikan bahwa setiap fitur utama, mulai dari autentikasi pengguna hingga proses pembayaran dan notifikasi, berfungsi sesuai kebutuhan fungsional yang telah diidentifikasi pada tahap awal penelitian.

Result and Discussion

1. Perancangan Arsitektur Sistem dan Basis Data

Pengembangan sistem menghasilkan rancangan arsitektur *e-commerce multi-tenant* yang mengintegrasikan antarmuka pengguna, logika bisnis, dan basis data dalam satu ekosistem aplikasi berbasis Next.js App Router. Pendekatan ini dipilih karena mampu mengintegrasikan proses *frontend* dan *backend* dalam satu repositori sehingga pengembangan, pemeliharaan, serta proses *deployment* menjadi lebih sederhana dibandingkan pendekatan yang memisahkan kedua komponen tersebut. Selain itu, pemanfaatan React Server Components dan API Routes mendukung pengelolaan proses autentikasi, transaksi, dan komunikasi data secara terintegrasi, sehingga sesuai dengan karakteristik sistem multi-tenant yang membutuhkan pengelolaan banyak pengguna dan tenant dalam satu platform (Srivastava et al., 2024).

Arsitektur yang diusulkan dirancang untuk mengakomodasi tiga aktor utama dalam sistem, yaitu administrator, *tenant*, dan *customer*, yang masing-masing memiliki fungsi dan hak akses berbeda. Meskipun seluruh pengguna memanfaatkan platform yang sama, setiap proses bisnis dijalankan secara independen melalui mekanisme pemisahan modul berdasarkan peran (*role-based routing*). Pendekatan ini tidak hanya meningkatkan modularitas aplikasi, tetapi juga mempermudah pengelolaan kode program serta pengembangan fitur baru tanpa memengaruhi modul lain (Naofal et al., 2022). Dengan demikian, arsitektur sistem mampu mendukung kebutuhan skalabilitas ketika jumlah tenant, pengguna, maupun transaksi terus meningkat.

Pemilihan Next.js App Router pada penelitian ini tidak hanya didasarkan pada kemudahan pengembangan aplikasi, tetapi juga pada kemampuan Server-Side Rendering (SSR) dalam mengurangi proses rendering di sisi klien sehingga waktu pemuatan halaman menjadi lebih efisien. Karakteristik tersebut sejalan dengan temuan Hanafi dkk. (Hanafi et al., 2024) yang menunjukkan bahwa penerapan SSR mampu meningkatkan performa aplikasi web dibandingkan pendekatan yang sepenuhnya mengandalkan proses rendering di sisi klien. Pada penelitian ini, keunggulan tersebut semakin relevan karena sistem harus menangani akses dari berbagai tenant dan pengguna secara bersamaan, sehingga efisiensi proses rendering berkontribusi terhadap stabilitas dan responsivitas aplikasi.

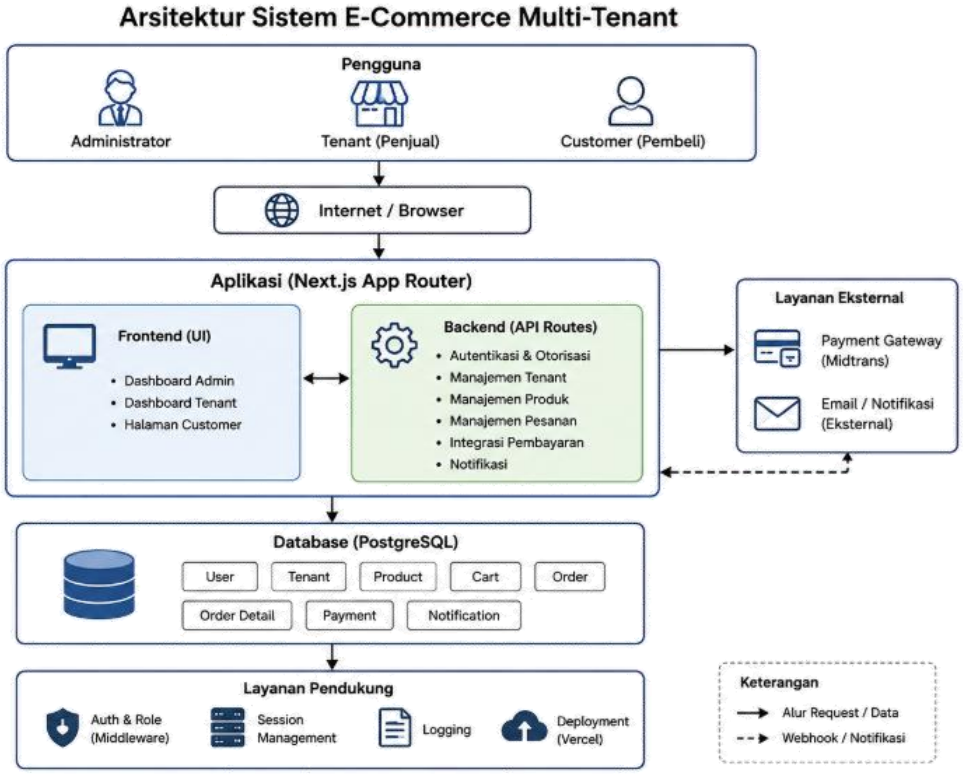

Figure 2. Arsitektur Sistem E-Commerce Multi-Tenant

Pada gambar tersebut ditunjukkan bahwa sistem dibangun menggunakan arsitektur berlapis (*layered architecture*) yang menghubungkan antarmuka pengguna dengan layanan aplikasi, basis data, dan layanan eksternal melalui mekanisme yang saling terintegrasi. Struktur direktori dipisahkan berdasarkan peran pengguna, yaitu administrator, *tenant*, dan *customer*, sehingga setiap modul memiliki tanggung jawab yang jelas serta dapat dikembangkan secara independen (Mulyadi, 2023). Pemisahan tersebut tidak hanya meningkatkan modularitas aplikasi, tetapi juga mempermudah proses pemeliharaan dan pengembangan fitur baru tanpa memengaruhi modul lainnya.

Sebagai pendukung arsitektur sistem, penelitian ini juga merancang basis data relasional yang terdiri atas delapan entitas utama, yaitu *User, Tenant, Product, Cart, Order, Order Detail, Payment, dan Notification*. Rancangan tersebut disusun untuk mendukung seluruh proses bisnis mulai dari pengelolaan pengguna, pengelolaan toko, transaksi pembelian, hingga penyampaian notifikasi secara terintegrasi. Hubungan antarentitas dirancang agar setiap transaksi dapat ditelusuri secara konsisten serta menjaga integritas data selama proses operasional sistem berlangsung (Ahmed I. Saleh, Mohammed A. Fouad, 2014).

Perancangan basis data tidak hanya difokuskan pada penyimpanan informasi, tetapi juga pada kemampuan sistem dalam menjaga konsistensi hubungan antarentitas selama proses bisnis berlangsung. Setiap entitas dirancang untuk merepresentasikan aktivitas yang berbeda, mulai dari identitas pengguna, pengelolaan tenant, katalog produk, proses transaksi, hingga mekanisme pembayaran dan notifikasi (Fatman et al., 2023). Dengan pendekatan tersebut, setiap perubahan data yang terjadi pada satu proses dapat direpresentasikan secara konsisten pada proses lainnya sehingga mengurangi potensi redundansi maupun inkonsistensi data.

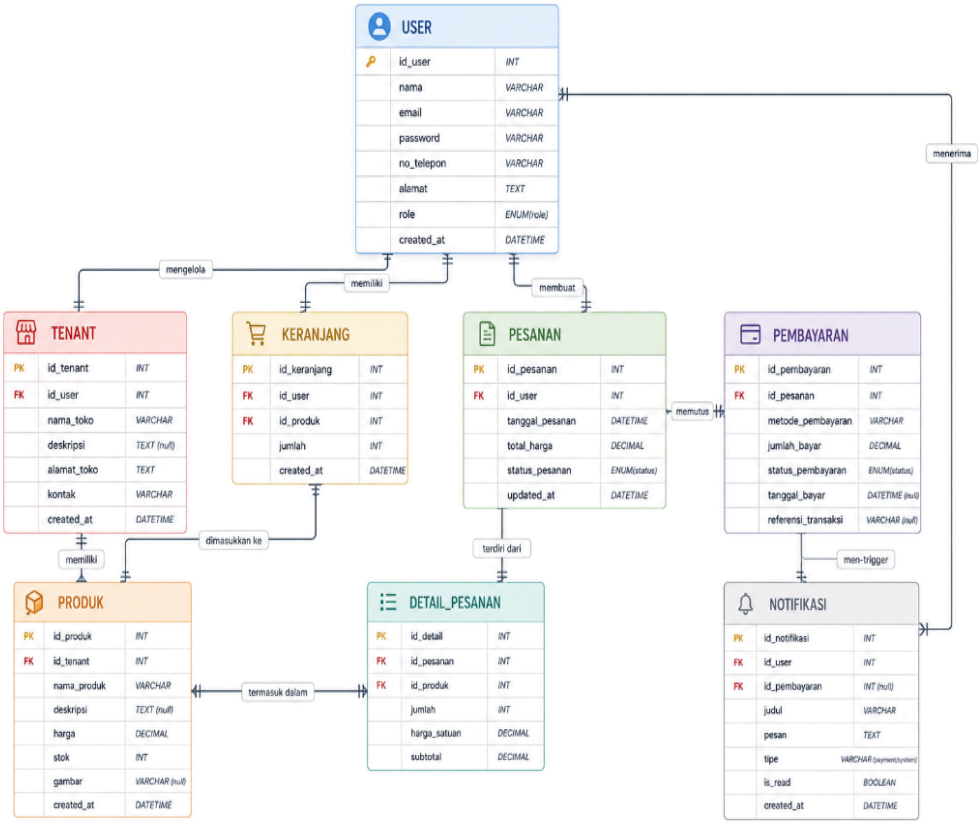

Figure 3. Entity Relationship Diagram (ERD) Sistem

Entity Relationship Diagram (ERD) pada gambar memperlihatkan bahwa setiap tenant memiliki ruang pengelolaan data yang terpisah meskipun seluruh tenant berada dalam satu basis data yang sama. Pendekatan ini merupakan karakteristik utama sistem *multi-tenant*, yaitu memungkinkan penggunaan sumber daya secara bersama tanpa menghilangkan isolasi data antar-tenant (Jadhav, Bandgar, Fattepurkar, 2014) Selain meningkatkan efisiensi pengelolaan basis data, rancangan relasi antarentitas juga mendukung proses autentikasi, pengelolaan produk, transaksi, pembayaran, dan notifikasi secara terintegrasi sehingga aliran data antarproses menjadi lebih konsisten.

Hasil perancangan arsitektur dan basis data menunjukkan bahwa integrasi Next.js App Router, konsep multi-tenant, dan rancangan basis data relasional membentuk fondasi sistem yang modular, mudah dikembangkan, serta mampu mendukung kebutuhan operasional marketplace modern. Temuan ini memperluas hasil penelitian sebelumnya yang umumnya hanya berfokus pada aspek framework, basis data, atau arsitektur secara terpisah, dengan mengintegrasikan seluruh komponen tersebut ke dalam satu rancangan sistem yang terpadu sebagai dasar implementasi pada tahap berikutnya (Mehmannavaz, 2013).

2. Implementasi Mekanisme Multi-Tenant

Rancangan arsitektur dan basis data yang telah disusun pada tahap sebelumnya kemudian diimplementasikan menjadi sistem *e-commerce multi-tenant berbasis* Next.js App Router. Implementasi dilakukan dengan menerapkan mekanisme autentikasi, otorisasi berbasis peran (*role-based access control*), serta pemisahan antarmuka pengguna sesuai dengan peran administrator, tenant, dan customer. Pendekatan ini bertujuan memastikan bahwa setiap pengguna hanya dapat mengakses fungsi sesuai kewenangannya sehingga keamanan sistem, konsistensi proses bisnis, dan isolasi data antar-tenant tetap terjaga.Implementasi sistem dirancang agar setiap proses bisnis berlangsung secara terintegrasi, mulai dari autentikasi pengguna, pengelolaan produk, transaksi pemesanan, hingga proses pembayaran (Sharma, 2025). Seluruh alur tersebut dibangun berdasarkan konsep *role-based access control* (RBAC) sehingga setiap aktivitas pengguna diproses sesuai hak akses yang dimiliki. Pendekatan ini tidak hanya meningkatkan keamanan sistem, tetapi juga menyederhanakan alur interaksi pengguna karena setiap aktor hanya berinteraksi dengan fitur yang relevan terhadap tugasnya. Dengan demikian, implementasi sistem mampu mendukung operasional marketplace yang melibatkan banyak tenant secara lebih terstruktur, efisien, dan mudah dikembangkan (Haris et al., 2025).

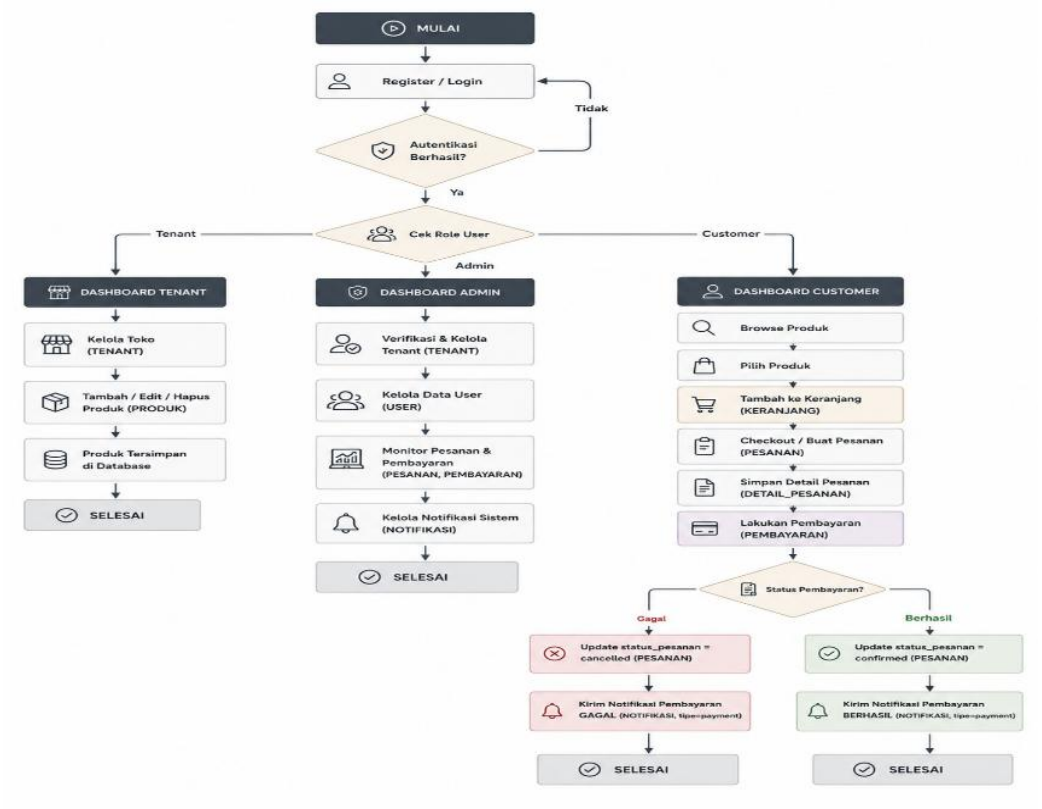

Figure 4. *Flowchart* Alur Sistem E-Commerce Multi-Tenant

Flowchart tersebut menggambarkan alur operasional sistem yang dimulai dari proses autentikasi pengguna hingga penyelesaian transaksi. Setelah pengguna berhasil melakukan login, sistem akan melakukan verifikasi peran untuk menentukan hak akses yang dimiliki. Administrator diarahkan ke dashboard pengelolaan platform, tenant menuju dashboard pengelolaan toko dan produk, sedangkan customer diarahkan ke halaman pembelian. Pemisahan alur berdasarkan peran tersebut merupakan implementasi konsep *multi-tenant*, di mana seluruh pengguna memanfaatkan platform yang sama tetapi hanya memperoleh akses terhadap fungsi dan data yang sesuai dengan kewenangannya (Muhammad Dedi Irawan, 2024).

Penerapan mekanisme tersebut tidak hanya meningkatkan keamanan akses sistem, tetapi juga menjaga independensi setiap tenant dalam mengelola aktivitas bisnisnya. Administrator memiliki kewenangan untuk mengelola pengguna, melakukan verifikasi tenant, memantau transaksi, serta mengelola notifikasi sistem. Di sisi lain, tenant hanya dapat mengelola toko, produk, dan pesanan miliknya sendiri, sedangkan customer difokuskan pada aktivitas pencarian produk, pengelolaan keranjang, proses checkout, hingga pembayaran. Pembagian hak akses tersebut mengurangi potensi akses tidak sah terhadap data tenant lain sekaligus menjaga konsistensi proses bisnis dalam lingkungan *multi-tenant*, sebagaimana dikemukakan oleh (Jadhav, Bandgar, Fattepurkar, 2014).

Selain mendukung keamanan sistem, pemisahan antarmuka berdasarkan peran juga meningkatkan kemudahan penggunaan (*usability*). Setiap pengguna hanya berinteraksi dengan informasi dan fungsi yang relevan terhadap tugasnya sehingga kompleksitas antarmuka dapat dikurangi dan proses operasional menjadi lebih efisien. Temuan ini sejalan dengan penelitian (Vázquez-ingelmo et al., 2019) yang menyatakan bahwa dashboard berbasis peran mampu meningkatkan efektivitas penyajian informasi, serta didukung oleh (Aini et al., 2020) yang menunjukkan bahwa penyederhanaan antarmuka sesuai kebutuhan pengguna berkontribusi terhadap peningkatan pengalaman pengguna (*user experience*).

Implementasi sistem juga menunjukkan bahwa pemanfaatan Next.js App Router mendukung pengelolaan autentikasi, navigasi, dan pemisahan modul aplikasi secara lebih terstruktur dibandingkan pendekatan konvensional yang memisahkan *frontend* dan *backend* ke dalam proyek yang berbeda. Integrasi tersebut mempermudah proses pengembangan sekaligus menjaga konsistensi komunikasi antara antarmuka pengguna, logika bisnis, dan basis data. Hasil ini memperkuat temuan (Hanafi et al., 2024) bahwa pemanfaatan fitur modern pada Next.js mampu meningkatkan efisiensi pengembangan aplikasi web, dan pada penelitian ini penerapannya diperluas untuk mendukung implementasi sistem *e-commerce multi-tenant.* Secara keseluruhan, implementasi sistem membuktikan bahwa konsep

multi-tenant tidak hanya diterapkan pada rancangan arsitektur, tetapi juga diwujudkan melalui mekanisme autentikasi, otorisasi, pemisahan hak akses, dan pengelolaan antarmuka berbasis peran. Implementasi tersebut menghasilkan alur operasional yang lebih terstruktur, menjaga keamanan akses antar-pengguna, serta mendukung pengelolaan marketplace dengan banyak tenant secara efisien (Abdillah et al., 2025).

3. Implementasi Integrasi Payment Gateway

Untuk mendukung proses transaksi secara elektronik, sistem mengintegrasikan layanan Midtrans sebagai *payment gateway* menggunakan Snap API dan Core API. Integrasi ini memungkinkan customer melakukan pembayaran melalui berbagai metode pembayaran yang didukung Midtrans, sedangkan sistem secara otomatis memproses status transaksi tanpa memerlukan konfirmasi manual. Pendekatan tersebut dirancang agar proses pembayaran dapat berlangsung lebih cepat, aman, dan terintegrasi dengan keseluruhan proses bisnis pada sistem *e-commerce multi-tenant* (Rohunen et al., 2014). Implementasi layanan pembayaran dilakukan melalui mekanisme webhook, yaitu metode komunikasi yang memungkinkan Midtrans mengirimkan notifikasi secara otomatis kepada server aplikasi setiap terjadi perubahan status transaksi. Berbeda dengan mekanisme *polling* yang mengharuskan aplikasi melakukan pengecekan status pembayaran secara berkala, webhook bekerja secara *event-driven* sehingga informasi pembayaran dapat diterima secara *real-time*. Pendekatan ini tidak hanya mengurangi beban komunikasi antara aplikasi dan layanan pembayaran, tetapi juga meminimalkan keterlambatan pembaruan status transaksi sehingga konsistensi data pada sistem tetap terjaga (Revi Angeli Siahaan, 2024).

Selain memperbarui status pembayaran secara otomatis, sistem juga menerapkan mekanisme verifikasi menggunakan SHA-512 signature yang disediakan Midtrans sebelum setiap notifikasi diproses. Verifikasi tersebut bertujuan memastikan bahwa data yang diterima benar-benar berasal dari layanan pembayaran sehingga keamanan komunikasi antar sistem tetap terjamin. Setelah proses verifikasi berhasil dilakukan, status pembayaran akan diperbarui secara otomatis pada basis data dan informasi transaksi diteruskan ke dashboard sesuai peran pengguna (Sammir & Hamdi, 2025). Dengan demikian, administrator, tenant, maupun customer memperoleh informasi pembayaran yang sama secara konsisten tanpa memerlukan intervensi manual.

Implementasi ini menunjukkan bahwa integrasi *payment gateway* tidak hanya berfungsi sebagai media pembayaran, tetapi juga sebagai bagian dari mekanisme otomatisasi proses bisnis. Integrasi webhook dengan dashboard berbasis peran memungkinkan sinkronisasi informasi transaksi berlangsung secara lebih efisien dibandingkan pendekatan konvensional yang masih bergantung pada konfirmasi manual. Temuan ini memperluas penelitian (Maulana et al., 2022) yang berfokus pada integrasi Midtrans untuk otomatisasi pembayaran, karena pada penelitian ini mekanisme webhook juga dihubungkan dengan pengelolaan multi-tenant, sistem notifikasi, serta dashboard terpusat dalam satu arsitektur aplikasi.

Table 2. Perbandingan Karakteristik Penelitian dengan Penelitian Terdahulu

Aspek	**Penelitian Ini**	**Kusnadi & Putra** (Kusnadi & Putra, 2024)	**Maulana dkk.** (Maulana et al., 2022)
Framework	Next.js 16 (App Router)	CodeIgniter 4	Laravel
Metode pengembangan	Prototype	Waterfall (SDLC)	Tidak disebutkan eksplisit
Arsitektur frontend-backend	Terpadu (full-stack, satu proyek)	Terpisah (MVC monolitik)	Terpisah
Rendering	SSR/SSG (Next.js)	Server-side klasik (CI4)	Server-side klasik (Blade)
Payment gateway	Midtrans (Snap + Core API + webhook)	Tidak dibahas	Midtrans (API)
Jumlah peran pengguna	3 (admin, tenant, customer)	1-2 (umumnya admin-user)	Tidak dispesifikasi multi-tenant
Isolasi data multi-tenant	Ya (middleware filter id_tenant)	Tidak relevan (single-tenant)	Tidak relevan
Notifikasi real-time	Ya	Tidak dibahas	Tidak dibahas

Aspek	Penelitian Ini	**Kusnadi & Putra** (Kusnadi & Putra, 2024)	**Maulana dkk.** (Maulana et al., 2022)
Pengukuran performa non-fungsional	Ya (PageSpeed Insights, kuantitatif)	Tidak dibahas	Tidak dibahas

Dengan demikian menunjukan bahwa penelitian ini memiliki karakteristik yang lebih komprehensif dibandingkan penelitian terdahulu karena mengintegrasikan Next.js App Router, arsitektur multi-tenant, layanan pembayaran Midtrans berbasis webhook, pengelolaan dashboard berdasarkan peran pengguna, serta evaluasi performa menggunakan Google PageSpeed Insights dalam satu sistem yang terpadu. Penelitian Kusnadi dan Putra (Kusnadi & Putra, 2024) berfokus pada pengembangan e-commerce menggunakan *CodeIgniter* dengan pendekatan arsitektur konvensional, sedangkan Maulana dkk. (Maulana et al., 2022) menitikberatkan pada implementasi Midtrans tanpa membahas aspek multi-tenant maupun evaluasi performa aplikasi. Oleh karena itu, kontribusi utama penelitian ini tidak terletak pada penggunaan satu teknologi tertentu, melainkan pada sintesis berbagai teknologi modern menjadi satu model implementasi *e-commerce multi-tenant* yang terintegrasi, sehingga dapat dijadikan referensi dalam pengembangan marketplace mandiri bagi institusi pendidikan maupun komunitas UMKM.

4. Validasi dan Evaluasi Sistem

Setelah seluruh tahapan implementasi selesai dilakukan, sistem divalidasi melalui pengujian fungsional dan evaluasi performa untuk memastikan bahwa seluruh kebutuhan pengguna telah terpenuhi. Pengujian fungsional dilakukan menggunakan metode *Black Box* Testing karena berfokus pada kesesuaian keluaran sistem terhadap spesifikasi kebutuhan tanpa memperhatikan struktur kode program. Sementara itu, evaluasi performa dilakukan menggunakan *Google PageSpeed Insights* untuk mengukur kualitas non-fungsional aplikasi, meliputi performa, aksesibilitas, praktik terbaik (*best practices*), dan optimasi mesin pencari (*SEO*) (Ari Yogi Prasetyo, 2024). Kombinasi kedua pendekatan tersebut memberikan gambaran yang komprehensif mengenai kualitas sistem, baik dari aspek fungsional maupun non-fungsional.

Table 3. Hasil Pengujian *Black Box* Sistem *E-Commerce Multi-Tenant*

No	Fitur yang Diuji	Skenario Uji	Hasil yang Diharapkan	Status
1	*Registrasi & Login*	Pengguna mendaftar dan *login* dengan kredensial valid	Akun terbuat, diarahkan ke *dashboard* sesuai peran	Berhasil
2	Verifikasi *Tenant* (*Admin*)	Admin memverifikasi permohonan *tenant* baru dari *dashboard* admin	Status *tenant* aktif, *tenant* dapat *login* dan kelola toko	Berhasil
3	Kelola Data *User* (*Admin*)	*Admin* mengakses, mencari, dan memfilter daftar pengguna berdasarkan peran	Data *user* ditampilkan, filter berjalan sesuai peran	Berhasil
4	Tambah / Edit Produk (*Tenant*)	*Tenant* menambahkan produk baru dengan gambar dan stok	Produk tersimpan di tabel produk, tampil di katalog	Berhasil
5	Keranjang Belanja (*Customer*)	*Customer* menambahkan beberapa produk dari *tenant* berbeda	Item tercatat di keranjang dengan data *tenant* dan produk	Berhasil
6	*Checkout* & Pembayaran	*Customer checkout* dan membayar melalui antarmuka Midtrans Snap	pesanan & detail_pesanan terbuat, status pending	Berhasil
7	Konfirmasi *Webhook* Midtrans	Midtrans mengirimkan notifikasi pembayaran berhasil ke endpoint *webhook*	Status pesanan berubah *confirmed*, notifikasi dikirim	Berhasil
8	Monitoring Pesanan (*Admin*)	Admin memantau seluruh pesanan dan status pembayaran secara *real-time*	Data pesanan & pembayaran tampil lengkap di *dashboard* admin	Berhasil

No	Fitur yang Diuji	Skenario Uji	Hasil yang Diharapkan	Status
9	Isolasi Data *Tenant*	*Tenant* A mencoba mengakses API data produk *Tenant* B	Permintaan ditolak dengan kode HTTP 403 *Forbidden*	Berhasil
10	Notifikasi Sistem	*Customer* & *tenant* menerima notifikasi setelah transaksi selesai	Entri notifikasi terbuat dengan tipe dan is_read = false	Berhasil

Seluruh skenario pengujian berhasil dijalankan sesuai dengan fungsi yang dirancang, mulai dari proses registrasi dan autentikasi pengguna, verifikasi tenant, pengelolaan produk, transaksi pembelian, integrasi pembayaran melalui Midtrans, mekanisme webhook, isolasi data antar-tenant, hingga pengiriman notifikasi sistem. Keberhasilan seluruh skenario tersebut menunjukkan bahwa integrasi antara arsitektur aplikasi, basis data, serta layanan pembayaran mampu mendukung keseluruhan proses bisnis e-commerce multi-tenant secara konsisten tanpa ditemukan kesalahan fungsional selama proses pengujian (Baehaqi et al., 2023).

Temuan tersebut sekaligus menunjukkan efektivitas penerapan metode *Prototype* dalam proses pengembangan sistem. Melalui tahapan *Listen to Customer*, *Build/Revise Mock-up*, dan *Customer Test Drives Mock-up*, kebutuhan pengguna dapat divalidasi secara bertahap sehingga penyempurnaan sistem dilakukan sebelum implementasi akhir. Pendekatan iteratif tersebut mengurangi potensi ketidaksesuaian antara kebutuhan pengguna dan sistem yang dikembangkan, sehingga seluruh fitur utama dapat berfungsi sesuai kebutuhan. Hasil ini sejalan dengan Pressman dan Maxim (Roger Pressman, 2020) yang menyatakan bahwa metode *Prototype* efektif diterapkan pada pengembangan perangkat lunak yang memiliki kebutuhan dinamis dan memerlukan keterlibatan pengguna secara berkelanjutan.

Table 4. Hasil Pengukuran Performa Sistem Menggunakan *Google PageSpeed Insights*

Metrik	Nilai
Skor Performa	92 / 100
First Contentful Paint	0,3 detik
Largest Contentful Paint	0,8 detik
Total Blocking Time	20 md
Cumulative Layout Shift	0
Speed Index	3,1 detik
Skor Aksesibilitas	100 / 100
Skor Praktik Terbaik	100 / 100
Skor SEO	100 / 100

Evaluasi performa menunjukkan bahwa sistem memperoleh skor performa sebesar 92/100, disertai nilai 100/100 pada aspek aksesibilitas, praktik terbaik (*best practices*), dan optimasi mesin pencari (*SEO*). Selain itu, metrik First Contentful Paint sebesar 0,3 detik, Largest Contentful Paint sebesar 0,8 detik, Total Blocking Time sebesar 20 ms, serta Cumulative Layout Shift bernilai 0 mengindikasikan bahwa halaman mampu dimuat dengan cepat, responsif, dan stabil selama proses interaksi pengguna. Capaian tersebut menunjukkan bahwa sistem telah memenuhi indikator kualitas aplikasi web modern yang berorientasi pada pengalaman pengguna (Mathew, 2025).

Pencapaian tersebut dipengaruhi oleh pemanfaatan Server-Side Rendering (SSR), React Server Components, serta mekanisme App Router pada Next.js yang mampu mengurangi beban pemrosesan JavaScript di sisi klien. Berbeda dengan pendekatan *server-side rendering* konvensional pada framework seperti Laravel atau CodeIgniter yang menghasilkan halaman secara penuh pada setiap permintaan, Next.js mengoptimalkan proses rendering melalui kombinasi komponen server dan mekanisme *routing* modern sehingga proses pemuatan halaman menjadi lebih efisien. Kondisi ini berkontribusi terhadap rendahnya waktu *First Contentful Paint* dan *Largest Contentful Paint*, sekaligus menjaga stabilitas tampilan halaman yang tercermin dari nilai *Cumulative Layout Shift* sebesar nol. Temuan tersebut memperkuat penelitian Hanafi dkk. (Hanafi et al., 2024) yang menunjukkan bahwa pendekatan rendering modern pada Next.js mampu meningkatkan performa aplikasi web, dan pada penelitian ini keunggulan tersebut dibuktikan pada implementasi sistem e-commerce multi-tenant yang mengintegrasikan dashboard berbasis peran, layanan pembayaran, dan basis data terpadu.

Secara keseluruhan, hasil validasi dan evaluasi menunjukkan bahwa sistem tidak hanya memenuhi seluruh kebutuhan fungsional, tetapi juga memiliki kualitas non-fungsional yang baik dari aspek performa, aksesibilitas, praktik terbaik, dan optimasi SEO. Kombinasi metode *Prototype*, framework Next.js App Router, arsitektur multi-tenant, serta integrasi Midtrans berbasis webhook menghasilkan sistem yang mampu mendukung proses bisnis marketplace secara efektif sekaligus memberikan pengalaman pengguna yang baik (Firdhayanti et al., 2023). Dengan demikian, penelitian ini tidak hanya menghasilkan sistem yang berjalan sesuai spesifikasi, tetapi juga memberikan bukti empiris bahwa integrasi berbagai teknologi modern mampu meningkatkan kualitas implementasi *e-commerce multi-tenant.*

Conclusion

Penelitian ini menunjukkan bahwa implementasi arsitektur Next.js App Router pada sistem informasi *e-commerce multi-tenant* mampu mengintegrasikan proses frontend dan backend dalam satu aplikasi secara efektif. Sistem yang dikembangkan berhasil mendukung pengelolaan tiga peran pengguna, yaitu administrator, tenant, dan *customer*, dengan mekanisme isolasi data antar-tenant yang konsisten. Selain itu, integrasi payment gateway Midtrans melalui mekanisme webhook memungkinkan pembaruan status pembayaran berlangsung secara otomatis dan *real-time*, sehingga meningkatkan efisiensi proses transaksi. Hasil pengujian fungsional dan evaluasi performa menunjukkan bahwa sistem telah memenuhi kebutuhan pengguna serta memiliki kualitas implementasi yang baik sebagai platform marketplace berbasis web.

Kontribusi ilmiah penelitian ini terletak pada penyusunan model implementasi *e-commerce multi-tenant* berbasis Next.js App Router yang mengintegrasikan arsitektur aplikasi, pengelolaan basis data, *dashboard* berbasis peran, serta layanan pembayaran dalam satu sistem yang terpadu. Model tersebut memperkaya kajian pengembangan sistem e-commerce dengan menunjukkan bahwa pendekatan *full-stack* berbasis Next.js dapat diterapkan sebagai alternatif arsitektur modern untuk pengembangan marketplace multi-tenant. Dari sisi praktis, model ini dapat dijadikan referensi bagi pengembangan marketplace mandiri pada UMKM, koperasi, maupun institusi pendidikan yang memerlukan sistem terintegrasi dengan biaya pengembangan yang lebih efisien. Meskipun demikian, penelitian ini masih terbatas pada implementasi dalam satu lingkungan aplikasi (*monolithic full-stack*) dan belum mengevaluasi performa sistem pada skala pengguna yang sangat besar maupun lingkungan *cloud* terdistribusi. Oleh karena itu, penelitian selanjutnya dapat mengembangkan arsitektur berbasis *microservices*, mengintegrasikan Elasticsearch untuk meningkatkan kemampuan pencarian produk, menambahkan modul logistik secara *real-time*, serta melakukan pengujian skalabilitas pada beban pengguna yang lebih masif.

References

Abd-elfatah Tamimi, Thamer AL-Rawashdeh, H. abutaleb. (2021). Empirical Study of Most Popular PHP Framework. *Journal of International Conference on Information Technology*, *10*(1).

Abdillah, A. M., Pinandito, A., & Pramono, D. (2025). Analisis Perbandingan Performa Drizzle ORM , Prisma ORM , dan Raw SQL dalam Web Service Berbasis RESTful API dan JavaScript. *Jurnal Pengembangan Teknologi Informasi Dan Ilmu Komputer*, *9*(12), 1–10.

Ahmed I. Saleh, Mohammed A. Fouad, M. A.-E. (2014). A Hybrid Multi-Tenant Database Schema for Multi- Level Quality of Service. *International Journal of Advanced Computer Science and Applications*, *5*(11), 132–139.

Aini, Q., Rahardja, U., Aisyah, E. S., & Khoirunisa, A. (2020). Performa Kinerja Admin Layanan Keuangan Mahasiswa Menggunakan Dashboard Pada Web Based Accounting Online. *Jurnal Ilmiah Ilmu Komputer*, *15*(1).

Ari Yogi Prasetyo, G. S. (2024). Sistem informasi manajemen ticketing event dengan payment gateway berbasis Node.js dan Next.js. *Journal of Information System and Application Development*, *2*(2), 100–110. https://doi.org/10.26905/jisad.v2i2.12982

Baehaqi, A., Bashit, M. S., Indrajit, R. E., & Kurniawan, R. D. (2023). *FRONT END LEARNING MANAGEMENT SYSTEM DEVELOPMENT USING THE PENGEMBANGAN FRONT END LEARNING MANAGEMENT SYSTEM*. *4*(4), 899–911.

Falih, N. (2019). Analisa Kebutuhan E-commerce untuk UKM Menggunakan Goal- Oriented Requirement Engineering (GORE). *JURNAL INFORMATIK*, *15*, 11–20.

Fatman, Y., Nafisah, N. K., Bendoro, P., & Pambudi, J. (2023). Implementasi Payment Gateway dengan Menggunakan Midtrans pada Website UMKM Geberco. *Jurnal KomtekInfo*, *10*(2), 64–72. https://doi.org/10.35134/komtekinfo.v10i2.364

Firdhayanti, A., Taufik, T., & Bachry, B. (2023). *User Acceptance Testing through Blackbox Evaluation for Corn Distribution Information System*. *6*(2). https://doi.org/10.32877/bt.v6i2.1065

Hanafi, R., Haq, A., & Agustin, N. (2024). Comparison of Web Page Rendering Methods Based on Next . js Framework Using Page Loading Time Test. *Jurnal TEKNIKA*, *13*(1), 102–108. https://doi.org/10.34148/teknika.v13i1.769

Haris, M., Mahariq, Z., Arnaout, W., Muiz, M. Y., Haris, M., Mahariq, Z., Arnaout, W., & Muiz, M. Y. (2025). Enhancing Retail Efficiency: Development and Evaluation of a Self Checkout System. *Journal of Preprints*, 1–10. https://doi.org/10.20944/preprints202504.0761.v1

Harshit Mishra, Harshit Shukla, E. S. K. D. (2024). Navigating the Digital Marketplace : Insights into E-commerce Development with MERN Stack. *International Journal for Research in Applied Science & Engineering Technology*, *12*(5).

Jadhav, Bandgar, Fattepurkar, B. (2014). An Approach for Development of Multitenant Application as SaaS Cloud. *International Journal of Computer Applications*, *106*(14), 15–18.

Jain, U. (2024). Challenges in maintaining data privacy and compliance in multi-tenant cloud environments. *Journal of Quantum Science and Technology*, *1*(4), 154–167.

Kabbedijk, J. (2014). *Variability in Multi-Tenant Enterprise Software* (1st ed.).

Kenneth C. Laudon, J. P. L. (2014). *Management Information Systems Managing the Digital Firm* (S. Wall (ed.); 13th ed.).

Kusnadi, Y., & Putra, D. W. (2024). E-Commerce Berbasis Website pada UMKM Menggunakan Framework Codeigniter 4 (Studi Kasus: Toko Wakuteka). *Jurnal Teknlogi Informatika Dan Komputer MH. Thamrin*, *10*(1), 261–275.

Lipiński, A., & Pańczyk, B. (2023). Performance optimization of web applications using Qwik. *Journal of Computer Sciences Institute*, *28*(7), 197–203.

Maggenta, M. Z., Sianturi, R. S., & Kharisma, A. P. (2022). Perancangan User Experience Website Marketplace dan Pemetaan Hasil Pertanian menggunakan Metode Five Planes. *Jurnal Pengembangan Teknologi Informasi Dan Ilmu Komputer*, *6*(7).

Mathew, P. (2025). Front-End Performance Optimization for Next-Generation Digital Services. *Journal of Computer Science and Technology Studies*, *7*(4), 993–1000. https://doi.org/10.32996/jcsts

Maulana, M., Aditya, Z., Hayuhardhika, W., Putra, N., & Arwani, I. (2022). Pengembangan Sistem Informasi E-Commerce dengan Pemanfaatan API Midtrans menggunakan Framework Laravel (Studi Kasus : Byboot . id). *Jurnal Pengembangan Teknologi Informasi Dan Ilmu Komputer*, *6*(8), 3899–3906.

Mehmannavaz, V. (2013). CONCEPTS AND APPLICATIONS OF MANAGEMENT INFORMATION SYSTEMS. *Arabian Journal of Business and Management Review*, *2*(6), 6–15.

Muhammad Dedi Irawan, R. M. (2024). E-Letter Design Using Prototype System Development Methodology. *Bigint Computing Journal*, *2*(1).

Mulyadi, D. N. S. (2023). WEB-BASED SPP PAYMENT INFORMATION SYSTEM WITH MIDTRANS PAYMENT GATEWAY. *Journal of Information Systems and Computer Science Prima*, *6*(2), 21–27.

Namira Syahputri, M. I. P. N. (2024). Application Development E-Commerce at HNI Products Sales is using The Agile Method. *Jurnal Sistem Informasi*, *13*(2), 643–655.

Naofal, N., Rifqi, M., Ulhaq, D., & Prianto, C. (2022). *Pembuatan Sistem Informasi E-Commerce pada Toko Az-Zahra Menggunakan Framework Laravel Development of E-Commerce Information System at Az-Zahra Shop Using Laravel Framework*. *1*(1), 95–106. https://doi.org/10.55123/jomlai.v1i1.176

Perillo, G. (2024). Digital and Circular Economy Models. *Journal of Civil Engineering and Environmental Sciences*, *10*(2), 61–62.

Pranoto, A. H. (2026). PEMBELAJARAN PAI BERBASIS KECERDASAN ARTIFISIAL: Strategi, Inovasi, dan Implementasi. In R. Raharjo, A. Sutiyono, & A. Syatori (Eds.), *Duta Sains Indonesia (DSI Press)* (pp. 1–90). Duta Sains Indonesia (DSI Press).

Prasetiyo, S. M., Gustiawan, R., Albani, F. R., Komputer, I., Informatika, T., Pamulang, U., & Selatan, T. (2024).

Analisis Pertumbuhan Pengguna Internet Di Indonesia. *Buletin Ilmiah Ilmu Komputer Dan Multimedia*, *2*(1), 65–71.

Revi Angeli Siahaan, R. A. S. (2024). ANALISIS PERBANDINGAN PAYMENT GATEWAY UNTUK SISTEM PEMBAYARAN BERBASIS APLIKASI DENGAN COMPARATIVE STUDY. *Jurnal Teknologi Informasi Dan Ilmu Komputer*, *11*(2). https://doi.org/10.25126/jtiik.20241127680

Roger Pressman, B. M. (2020). *Software Engineering: A Practitioner's Approach* (9th ed.). MCGraw Hill.

Rohunen, A., Liukkunen, K., Tulppo, T., & Chan, K. W. (2014). Implementing and Evaluating a Smart-M3 Platform-based Multi-vendor Micropayment System Pilot in the Context of Small Business. *Journal of Digital Information Management*, *12*(1).

Sammir, H., & Hamdi, K. (2025). Perancangan Lokapasar Terdistribusi Sebagai Publikasi Kalatog Produk UMKM. *JURNAL MANAJEMEN TEKNOLOGI INFORMATIKA*, *03*(2), 146–157.

Samuel Nugraha, D. W. C. (2023). Peningkatan Keamanan Database Pada Layanan Azure Melalui Metode Multi-Tenant Dengan Pendekatan Separate Database. *Jurnal Pendidikan Dan Teknologi Indonesia*, *3*(6), 233–240.

Sarah Nurkhalisah Ismail, Fajar Septian, Arisantoso3, H. S. (2026). PENGEMBANGAN SISTEM DASHBOARD PENJUALAN PADA UMKM BERBASIS WEBSITE (STUDI KASUS: TOKO WYLOZ). *URNAL REKAYASA INFORMASI SWADHARMA*, *6*(1).

Satria, R., Ahmad, I., & Gunawan, R. D. (2023). Rancang Bangun E-Marketplace Berbasis Mobile Untuk Meningkatkan Pelayanan Penjualan. *JURNAL INFORMATIKA DAN REKAYASA*, *4*(1), 89–95.

Sharma, R. K. (2025). Multi-Tenant Architectures in Modern Cloud Computing : A Technical Deep Dive. *International Journal of Scientific Research in Computer Science, Engineering and Information Technology*, *11*(1), 307–317.

Srivastava, H., Gupta, S., Anand, K., & Sharma, A. (2024). *PDF-Chat SaaS Platform Using MERN Stack*. *5*(1), 37–40.

Tadi, S. R. C. C. T. (2024). Modern Dynamic Rendering Techniques: Incremental Static Regeneration in React and Flutter. *International Journal of Science and Research*, *13*(5), 1874–1880.

Vázquez-ingelmo, A., Therón, R., & García-peñalvo, F. J. (2019). Information Dashboards and Tailoring Capabilities-A Systematic Literature Review. *Journal of IEEE Acsess*, *7*(4), 109673–109688. https://doi.org/10.1109/ACCESS.2019.2933472

Vyas, R. (2022). Comparative Analysis on Front-End Frameworks for Web Applications. *International Journal for Research in Applied Science & Engineering Technology*, *10*(J7).