

BAB II

TINJAUAN PUSTAKA

A. Kajian Teoritis

1. *Mass Customization*

Mass customization merupakan pendekatan produksi yang bertujuan memenuhi kebutuhan individual pelanggan tanpa menghilangkan efisiensi yang menjadi ciri produksi massal. Dalam konteks manufaktur modern, pendekatan ini tidak hanya berarti menyediakan variasi produk yang banyak, tetapi juga menuntut adanya modularitas produk, kelincahan proses, integrasi teknologi digital, dan kemampuan perusahaan menerjemahkan kebutuhan pelanggan menjadi spesifikasi yang dapat diproduksi secara konsisten.

Bouchard et al., (2023) menegaskan bahwa implementasi *mass customization* pada usaha manufaktur kecil dan menengah memerlukan dukungan faktor seperti modular *design*, *agility*, *collaborative network*, dan *digital transformation*. Sejalan dengan itu, Roy et al., (2023) menunjukkan bahwa strategi *Industry 4.0* pada UKM manufaktur perlu dibangun secara bertahap dengan *lean*, *agility*, dan modular *design* sebagai prasyarat penting. Berdasarkan pandangan tersebut, *mass customization* menjadi landasan teoritis utama dalam penelitian ini karena sistem yang dikembangkan diarahkan untuk melayani pemesanan pakaian kustom secara personal, tetapi tetap terstruktur dan efisien.

2. Product Configuration dan *Product Configurator*

Product configuration adalah proses menerjemahkan kebutuhan pelanggan menjadi kombinasi atribut produk yang valid berdasarkan seperangkat aturan teknis dan bisnis. Dalam konteks *mass customization*, *configurator* tidak hanya berfungsi sebagai media pemilihan opsi, tetapi juga sebagai mekanisme yang memastikan kombinasi pilihan pengguna tetap sesuai dengan batasan produk yang dapat diproduksi. Melalui studi konsensus pakar berbasis metode Delphi–Régnier, Bouchard et al., (2026) menegaskan bahwa *product configuration* pada *mass customization* di UKM manufaktur perlu dibangun berdampingan dengan modular design serta disesuaikan dengan batasan operasional dan regulasi perusahaan. Studi tersebut juga mengidentifikasi kondisi kunci implementasi *product configuration*, meliputi struktur produk yang modular, aturan konfigurasi yang jelas, serta dukungan proses bisnis dan sistem informasi yang memadai.

Dalam ranah apparel, Uduwela et al., (2023) menempatkan *natural language-based product configurator* sebagai komponen penting dalam proses pengembangan produk baru (*new product development*) di industri *apparel*, khususnya untuk menjembatani kebutuhan pelanggan melalui pendekatan bahasa alami. Berdasarkan pandangan tersebut, *product configurator* pada penelitian ini dipahami sebagai inti sistem yang menghubungkan preferensi pelanggan dengan spesifikasi produk yang siap diproses lebih lanjut oleh admin dan bagian produksi.

3. *Visual Product Configurator* pada Industri Apparel

Pada industri *apparel*, keputusan pelanggan tidak hanya ditentukan oleh fungsi produk, tetapi juga oleh model, warna, detail komponen, dan persepsi visual terhadap hasil akhir. Karena itu, *configurator* pakaian harus mampu menyajikan pilihan yang mudah dipahami, interaktif, dan cukup jelas untuk membantu pengambilan keputusan. Yi et al., (2022) menunjukkan bahwa *interaction design* pada *product configurator* berpengaruh terhadap pengalaman personalisasi konsumen. Lu, (2026) menunjukkan bahwa sistem kustomisasi pakaian personal berbasis deep learning dapat meningkatkan interaktivitas dan akurasi hasil personalisasi, tetapi tetap memerlukan antarmuka yang memandu agar preferensi pengguna dapat tertangkap secara terarah, khususnya bagi pengguna non-ahli.

Temuan tersebut menunjukkan bahwa visual *product configurator* pakaian perlu dibangun dengan orientasi pada kemudahan interaksi, kejelasan visual, dan keterpaduan alur personalisasi. Oleh karena itu, visual *product configurator* pada penelitian ini diposisikan sebagai sarana interaksi yang membantu pelanggan memahami hasil kustomisasi sejak awal, sehingga keputusan desain tidak hanya bergantung pada komunikasi teks, tetapi juga pada representasi visual yang lebih mudah dipahami.

4. Teknik *2D Layering* dan Visualisasi Produk Berbasis Web

Teknik *2D layering* dalam penelitian ini dipahami sebagai penyusunan beberapa elemen visual dua dimensi menjadi satu tampilan produk yang utuh. Pada *configurator* pakaian, elemen-elemen tersebut dapat berupa badan pakaian, kerah, lengan, warna, dan desain tambahan dari pengguna. Secara teknis, pendekatan ini didukung oleh karakteristik visualisasi berbasis SVG pada web. Balla & Gede, (2025) membandingkan performa beberapa pustaka pemetaan berbasis web bersifat open source, dan menemukan bahwa kecepatan rendering data vektor sangat dipengaruhi oleh banyaknya elemen serta tingkat kerumitan data yang ditampilkan, bukan hanya oleh pilihan pustakanya saja. Karakteristik tersebut menegaskan bahwa pengelolaan kompleksitas rendering perlu diperhatikan pada aplikasi berbasis web agar interaksi tetap lancar ketika jumlah layer visual meningkat.

Temuan tersebut memberi dasar bahwa pendekatan *2D layering* cocok digunakan pada *configurator* pakaian berbasis web karena setiap komponen produk dapat direpresentasikan sebagai layer terpisah yang relatif ringan, mudah diperbarui, dan responsif terhadap interaksi pengguna. Dengan demikian, penggunaan *2D layering* relevan untuk menghadirkan visualisasi produk yang interaktif tanpa bergantung pada simulasi tiga dimensi yang lebih berat.

5. Konva.js

Implementasi teknik 2D layering pada penelitian ini dibangun di atas Konva.js, yaitu pustaka JavaScript yang memperluas elemen HTML5 Canvas dengan model pemrograman berorientasi objek. Peramban modern menyediakan beberapa model penggambaran grafis yang berbeda, yaitu berbasis DOM, berbasis kanvas, dan berbasis WebGL, yang masing-masing memiliki karakteristik kinerja tersendiri (Koren Ivančević et al., 2026). Pada model berbasis kanvas, bentuk yang telah digambar langsung berubah menjadi piksel dan tidak lagi memiliki identitas yang dapat diacu kembali, sehingga pengembang memerlukan lapisan abstraksi untuk melacak setiap objek. Konva.js menyediakan abstraksi tersebut dengan menyimpan setiap bentuk sebagai objek tersendiri, sehingga objek tetap dapat dipindahkan, diskalakan, diputar, dan diberi *event listener* tanpa perlu menggambar ulang seluruh tampilan secara manual.

Struktur hierarki Konva.js tersusun atas *Stage*, *Layer*, dan *Shape*, dengan setiap *Layer* dipetakan menjadi satu elemen kanvas tersendiri. Pemisahan tersebut memungkinkan bagian tampilan yang statis dan yang sering berubah ditempatkan pada lapisan yang berbeda, sehingga pembaruan pada satu lapisan tidak memaksa penggambaran ulang lapisan lainnya. Karakteristik ini selaras dengan kebutuhan visual product configurator pakaian, karena setiap komponen produk seperti badan pakaian, kerah, lengan, dan ornamen dapat direpresentasikan sebagai lapisan terpisah yang urutan penumpukannya diatur melalui properti *z-*

index. Ketika pelanggan mengubah satu pilihan konfigurasi, hanya lapisan yang bersangkutan yang diperbarui sehingga respons visual tetap ringan.

Pemilihan pendekatan berbasis kanvas didukung bukti empiris terkini. Koren Ivančević et al., (2026) melakukan pengukuran lintas perangkat dan lintas sistem operasi terhadap teknologi animasi web dan menemukan bahwa pendekatan berbasis DOM masih stabil pada seratus objek animasi, tetapi menurun secara nyata pada lima ratus objek, sedangkan penggambaran berbasis kanvas mampu mempertahankan kegunaannya pada jumlah objek yang jauh lebih besar. Temuan tersebut memperkuat alasan penggunaan pustaka kanvas dibandingkan manipulasi DOM langsung untuk visualisasi produk yang tersusun atas banyak lapisan.

Pemilihan pustaka penggambar grafis juga perlu mempertimbangkan batas kemampuan masing-masing pustaka. Balla & Gede, (2025) membandingkan kinerja empat pustaka JavaScript sumber terbuka dan menemukan bahwa tidak ada satu pustaka pun yang unggul pada seluruh skenario, karena peringkat kinerjanya berubah seiring bertambahnya jumlah objek yang digambar. Temuan tersebut menegaskan bahwa pemilihan pustaka harus disesuaikan dengan karakteristik beban kerja aplikasi yang dibangun. Pada penelitian ini, jumlah lapisan yang digambar secara bersamaan tergolong sedikit, yaitu sebatas komponen pakaian yang sedang dikonfigurasi, sehingga Konva.js dinilai memadai tanpa memerlukan penggambaran berbasis WebGL yang jauh lebih kompleks.

Kebutuhan akan visualisasi yang responsif tersebut bukan semata persoalan teknis. Zhan & Sheldrick, (2026) menemukan bahwa proses kustomisasi pakaian secara daring kerap membingungkan pelanggan ketika berlangsung tanpa pendampingan, sehingga simulasi visual yang interaktif berperan penting dalam membentuk keterlibatan dan respons emosional pelanggan terhadap produk yang dikonfigurasi. Dengan demikian, pemilihan Konva.js pada penelitian ini tidak hanya dilandasi pertimbangan kinerja, melainkan juga oleh kebutuhan menghadirkan umpan balik visual yang segera sebagai penopang pengalaman kustomisasi.

6. *State Management* pada Aplikasi Web Interaktif

State management merupakan konsep pengelolaan data dan perubahan antarmuka agar seluruh bagian aplikasi tetap sinkron ketika pengguna melakukan interaksi. Pada aplikasi *configurator*, *state* mencakup data jenis pakaian, opsi kustomisasi, warna, ukuran, jumlah pesanan, desain yang diunggah, hasil kalkulasi harga, hingga data *checkout*. Veeri, (2024) menunjukkan bahwa pemilihan solusi *state management* (misalnya Redux dan Zustand) memengaruhi performa rendering, kompleksitas boilerplate, konsumsi memori, dan skalabilitas aplikasi React modern.

Hal tersebut menegaskan bahwa *state management* bukan sekadar persoalan teknis implementasi, tetapi merupakan fondasi yang menentukan konsistensi aliran data dalam aplikasi interaktif. Dalam penelitian ini, *state management* menjadi dasar penting agar perubahan pilihan pengguna dapat

langsung tercermin pada visualisasi produk, ringkasan spesifikasi, kalkulasi harga, dan data transaksi secara *real-time*.

7. Object-Oriented Design (OOD)

Object-Oriented Design (OOD) merupakan pendekatan perancangan perangkat lunak yang memodelkan sistem berdasarkan objek, kelas, atribut, operasi, dan relasi antarobjek. Pendekatan ini sesuai digunakan pada sistem informasi yang memiliki banyak entitas saling berhubungan. Dalam penelitian ini, sistem visual *product configurator* pakaian memiliki beberapa objek utama, seperti *User*, *JenisPakaian*, *AsetProduk*, *KategoriKustomisasi*, *OpsiKustomisasi*, *AturanHarga*, *Pesanan*, *Pembayaran*, *LogStatusPesanan*, dan *SuratPerintahKerja*.

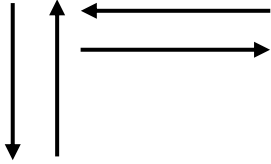
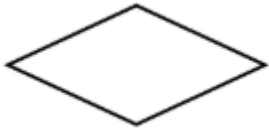
Rachmatika et al., (2025) memanfaatkan pendekatan OOAD untuk memetakan komponen-komponen sistem secara berorientasi objek, mulai dari aktor yang terlibat, jenis permintaan yang diajukan, hingga status penyelesaian setiap permohonan yang diproses sistem. Sumirat et al., (2023) menjelaskan bahwa rekayasa perangkat lunak mencakup tahapan analisis kebutuhan, perancangan, implementasi, pengujian, dan pemeliharaan yang menjadi dasar konseptual dalam pengembangan sistem informasi berbasis web. Letaw, (2022) menekankan bahwa metode rekayasa perangkat lunak modern mengintegrasikan pendekatan Agile, requirement engineering, dan pemodelan *UML* (khususnya class diagram dan sequence diagram) sebagai satu kesatuan praktik pengembangan sistem.

Melalui pendekatan tersebut, analisis dan desain sistem difokuskan pada pemodelan objek yang saling berinteraksi serta memiliki atribut dan perilaku yang jelas. Hal tersebut relevan dengan penelitian ini karena sistem yang dibangun memiliki objek-objek yang saling terhubung dalam proses pemesanan, konfigurasi produk, pembayaran, status pesanan, dan pembuatan SPK digital.

8. Flowchart

Flowchart merupakan bagan yang menggambarkan urutan langkah dan aliran logika suatu proses atau algoritma menggunakan simbol-simbol baku yang saling dihubungkan oleh garis alir, sehingga memudahkan pembacaan alur sistem secara sistematis (Nadiyah & Arifin, 2023). Dalam penelitian ini, flowchart digunakan untuk memodelkan alur proses utama sistem, yaitu mulai dari pengunjung memilih jenis pakaian pada halaman konfigurator, mengatur kustomisasi dan menyusun layer desain, melakukan *checkout*, mengunggah bukti pembayaran, hingga proses verifikasi pembayaran dan perubahan status pesanan oleh admin. Flowchart membantu memastikan logika alur pemesanan pada sistem berjalan runtut dan tidak ada langkah yang terlewat. Adapun simbol dan keterangan mengenai flowchart yang akan dijelaskan pada tabel.

Tabel 2. 1 Simbol *Flowchart*

Simbol	Keterangan
	Flow Direction Symbol Merupakan simbol yang digunakan untuk menghubungkan antara simbol yang satu dengan simbol yang lain. Bisa juga disebut <i>connecting line</i> .
	Terminator Symbol Yaitu simbol untuk permulaan (start) atau akhir (stop / end) dari suatu kegiatan.
	Processing Symbol Simbol yang menunjukkan pengolahan yang dilakukan oleh komputer.
	Symbol Decision Simbol pemilihan proses berdasarkan kondisi yang ada.
	Symbol Input-Output Simbol yang menyatakan proses input dan output tanpa tergantung dengan jenis peralatannya.

Sumber: Nadiyah & Arifin, (2023)

9. Unified Modeling Language (UML)

Pada penelitian ini, rancangan sistem dituangkan melalui *UML*, yakni notasi visual yang membantu tim pengembang memvisualisasikan

struktur dan perilaku sistem berorientasi objek sebelum diimplementasikan ke dalam kode program (Aurellia et al., 2025). Hidayati et al., (2023) menjelaskan bahwa pemodelan sistem menggunakan *UML* dapat diterapkan dalam perancangan sistem informasi berbasis web dengan diagram seperti use case diagram, activity diagram, sequence diagram, dan class diagram. *UML* membantu pengembang menggambarkan kebutuhan, alur proses, interaksi, dan struktur sistem sebelum masuk ke tahap implementasi.

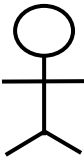
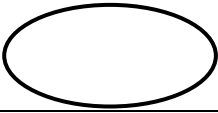
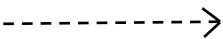
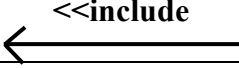
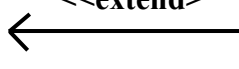
Dalam penelitian ini, *UML* digunakan sebagai alat bantu perancangan sistem berorientasi objek. Use case diagram digunakan untuk menggambarkan hubungan antara aktor dan fitur sistem. Activity diagram untuk menggambarkan alur aktivitas pada proses konfigurasi, *checkout*, dan pengelolaan pesanan. Sequence diagram untuk menggambarkan urutan interaksi antara aktor, antarmuka, controller, dan database, serta class diagram untuk menggambarkan struktur kelas, atribut, dan relasi antarobjek. Dengan *UML*, rancangan sistem dapat disusun secara sistematis sebelum diimplementasikan menggunakan Laravel, React/Inertia.js, dan MySQL.

a. Use Case Diagram

Use case diagram pada penelitian ini digunakan untuk memetakan siapa saja aktor yang berinteraksi dengan sistem beserta fungsi apa saja yang dapat mereka akses, tanpa perlu menjabarkan langkah teknis di baliknya (Rospricilia & Ma'ady, 2024).

Pada sistem ini terdapat dua aktor utama, yaitu pelanggan dan admin. Pelanggan dapat melakukan konfigurasi desain pakaian, menyusun layer, mengisi ukuran, melakukan *checkout*, mengunggah bukti pembayaran, serta melacak status pesanan; sedangkan admin dapat mengelola data master (jenis pakaian, aset produk, opsi kustomisasi, panduan ukuran, dan aturan harga), memverifikasi pembayaran, mengubah status pesanan, serta mencetak Surat Perintah Kerja (SPK) dalam bentuk PDF.

Tabel 2. 2 Simbol *Use Case Diagram*

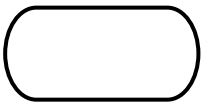
Simbol	Keterangan
	Aktor Mewakili peran orang, sistem yang lain, atau alat ketika berkomunikasi dengan use case.
	Use Case Abstraksi dan interaksi antara sistem dan aktor.
	Association Abstraksi dari penghubung antara aktor dengan use case.
	Generalisasi Menunjukkan spesialisasi aktor untuk dapat berpartisipasi dengan use case.
	Include Menunjukkan bahwa suatu use case seluruhnya merupakan fungsionalitas dari use case lainnya.
	Extend Menunjukkan bahwa suatu use case merupakan tambahan fungsional dari use case lainnya jika suatu kondisi terpenuhi.

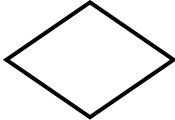
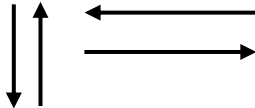
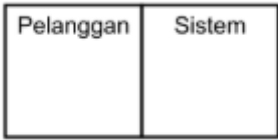
Sumber: Rospricilia & Ma'ady, (2024)

b. *Activity Diagram*

Activity diagram adalah diagram UML yang menggambarkan aliran aktivitas atau urutan kegiatan dalam suatu proses bisnis maupun proses sistem, termasuk percabangan kondisi dan aktivitas yang berjalan secara parallel (Tria & Lionie, 2025). Dalam penelitian ini, *activity* diagram digunakan untuk menggambarkan alur aktivitas pengguna ketika melakukan kustomisasi produk pada kanvas konfigurator hingga menyelesaikan pemesanan, serta alur aktivitas admin saat memverifikasi pembayaran dan memperbarui status pesanan yang setiap perubahannya dicatat pada *log* status pesanan.

Tabel 2. 3 Simbol *Activity Diagram*

Simbol	Keterangan
	<i>Initial Node</i> Menunjukkan titik awal (permulaan) dari alur aktivitas
	<i>Action</i> Menggambarkan satu langkah atau aksi yang dijalankan dalam alur aktivitas, misalnya “memvalidasi input”. Seluruh langkah pada diagram merupakan action.
	<i>Merge Node</i> Menggabungkan kembali beberapa alur alternatif yang berasal dari percabangan menjadi satu alur.
	<i>Activity Final Node</i> Menunjukkan titik akhir dari diagram aktivitas.

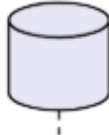
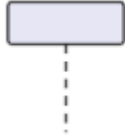
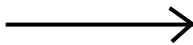
Simbol	Keterangan
	Decision Node Percabangan alur berdasarkan kondisi tertentu. Alur keluar diberi penjaga (<i>guard</i>), misalnya ya dan tidak
	Line Connector Digunakan untuk menghubungkan satu simbol dengan simbol lainnya.
	Swimlane (Partition) Membagi aktivitas berdasarkan aktor atau objek yang bertanggung jawab menjalankannya, misalnya Pelanggan dan Sistem

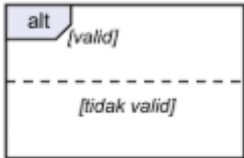
Sumber: Tria & Lionie, (2025)

c. *Sequence Diagram*

Sequence diagram merupakan diagram UML yang menggambarkan interaksi antarobjek dalam sistem secara berurutan berdasarkan waktu, dengan menampilkan pesan (*message*) yang dikirim dan diterima oleh setiap objek selama suatu proses berlangsung (Aurellia et al., 2025). Pada sistem ini, *sequence diagram* digunakan untuk memodelkan urutan interaksi ketika pelanggan menyimpan pesanan, yaitu mulai dari antarmuka React (melalui Inertia.js) mengirim permintaan ke *controller* pada Laravel, *controller* memproses dan menyimpan data ke basis data MySQL, kemudian mengirimkan respons kembali ke antarmuka pengguna.

Tabel 2. 4 Simbol *Sequence Diagram*

Simbol	Keterangan
	Actor Menggambarkan orang yang sedang berinteraksi dengan sistem.
	Objek / Partisipan Menggambarkan objek atau partisipan yang terlibat dalam interaksi, seperti halaman antarmuka, controller, dan model, beserta garis hidupnya (lifeline).
	Basis Data (Database) Menggambarkan media penyimpanan data yang menjadi tujuan penyimpanan maupun pengambilan data.
	Garis Hidup (lifeline) Menggambarkan keberadaan objek di sepanjang waktu selama interaksi berlangsung.
	Aktivasi (Focus of Control) Menggambarkan tempat mulai dan berakhirnya message.
	Pesan Panggilan Menggambarkan pemanggilan operasi atau pengiriman pesan dari satu objek ke objek lain.

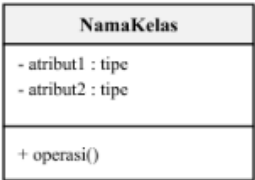
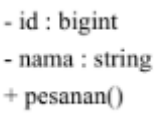
Simbol	Keterangan
	<i>Pesan Kembali (Return)</i> Menggambarkan nilai balik atau respons yang dikirim dari objek tujuan ke objek pemanggil.
	<i>Pesan ke Diri Sendiri</i> Menggambarkan objek yang memanggil operasinya sendiri, misalnya proses validasi input
	<i>Fragmen Kombinasi (alt)</i> Menggambarkan percabangan alur berdasarkan kondisi tertentu, seperti kondisi data valid dan tidak valid

Sumber: Aurellia et al., (2025)

d. *Class Diagram*

Struktur data pada sistem ini dimodelkan melalui class diagram, yang menampilkan kelas-kelas beserta atribut dan operasinya, sekaligus bagaimana kelas-kelas tersebut saling berelasi, baik melalui asosiasi, agregasi, maupun pewarisan (Ramdany et al., 2024). Pada penelitian ini, class diagram digunakan untuk menggambarkan struktur model data sistem yang diimplementasikan sebagai model Eloquent pada Laravel, seperti Pesanan, JenisPakaian, OpsiKustomisasi, KustomisasiPesanan, LayerDesainPesanan, UkuranPesanan, Pembayaran, LogStatusPesanan, dan SuratPerintahKerja, beserta relasinya misalnya, satu Pesanan memiliki banyak KustomisasiPesanan dan LayerDesainPesanan, serta berelasi dengan satu JenisPakaian.

Tabel 2. 5 Simbol *Class Diagram*

Simbol	Keterangan
	Class Himpunan objek yang memiliki atribut dan operasi yang sama. Digambarkan sebagai kotak tiga bagian, yaitu nama kelas, daftar atribut, dan daftar operasi
	Atribut dan Operasi Atribut ditulis dengan tanda minus (-) yang berarti private, sedangkan operasi ditulis dengan tanda plus (+) yang berarti public.
	Association Menghubungkan dua kelas yang saling berelasi. Arah panah menunjukkan kelas yang dirujuk oleh kelas asal.
	Multiplicity Menunjukkan jumlah objek yang terlibat dalam sebuah relasi, misalnya 1 (tepat satu), 0..*(nol atau lebih), dan 0..1 (paling banyak satu).
	Composition Relasi kepemilikan kuat, yaitu objek bagian tidak dapat berdiri sendiri tanpa objek induknya. Belah ketupat terisi diletakkan pada sisi kelas induk.

Sumber: Ramdany et al. (2024)

10. Basis Data MySQL

Dalam penelitian ini, basis data sistem dikelola menggunakan MySQL, sebuah RDBMS open source yang mengorganisasikan data ke dalam tabel-tabel saling terhubung dan diakses melalui SQL, sehingga proses penyimpanan maupun pengambilan data dapat dilakukan secara konsisten dan efisien (Karnasyah et al., 2025). Pendekatan serupa

diterapkan Duha et al., (2025) dalam perancangan basis data sistem informasi rumah sakit berbasis web dengan pemodelan ERD, CDM, dan PDM untuk memastikan struktur data yang konsisten dan relasi antar-entitas yang jelas.

Secara konseptual Mcfadyen, (2021) menjelaskan bahwa perancangan basis data relasional mencakup tahapan pemodelan Entity-Relationship, pemetaan ke tabel relasional, penerapan Data Definition Language (DDL) berbasis SQL, hingga normalisasi untuk menjaga integritas data. Pada penelitian ini, MySQL digunakan sebagai basis data sistem untuk menyimpan seluruh data, mulai dari data master produk dan opsi kustomisasi hingga data transaksi pesanan, pembayaran, dan log status, yang dikelola melalui Eloquent ORM dan migration pada framework Laravel.

11. Metode Pengembangan Sistem *Agile Scrum*

Agile Scrum merupakan metode pengembangan perangkat lunak yang bersifat iteratif dan inkremental. Metode ini menekankan proses pengembangan secara bertahap melalui pembagian pekerjaan ke dalam sprint, sehingga setiap fitur dapat dirancang, dibangun, diuji, dan dievaluasi secara berkelanjutan. Dalam praktiknya, Scrum banyak digunakan pada pengembangan perangkat lunak karena mampu membantu tim mengelola pekerjaan secara adaptif melalui peran, nilai, dan *event Scrum* yang saling mendukung keberhasilan proses pengembangan (Putrianasari et al., 2024).

Scrum memiliki beberapa komponen dan tahapan kerja utama, yaitu *product backlog*, *sprint planning*, *sprint backlog*, *daily scrum*, *sprint review*, dan *sprint retrospective*. *Product backlog* digunakan untuk menyusun kebutuhan dan prioritas fitur sistem. *Sprint planning* digunakan untuk merencanakan pekerjaan yang akan dilakukan pada satu periode *sprint*. *Sprint backlog* berisi daftar pekerjaan yang dipilih untuk diselesaikan dalam *sprint*. *Daily scrum* digunakan untuk memantau perkembangan pekerjaan, sedangkan *sprint review* dan *sprint retrospective* digunakan untuk mengevaluasi hasil serta memperbaiki proses pengembangan berikutnya (Abtokhi et al., 2023).

Dalam pengembangan sistem informasi, Dzaky & Kurniawan, (2023) menyoroti bahwa penerapan *Scrum* membantu tim membagi pekerjaan besar menjadi unit-unit *sprint* yang lebih terkendali, sekaligus memberi ruang bagi tim untuk menyesuaikan prioritas fitur dan merespons masukan pengguna secara berkelanjutan selama pengembangan sistem informasi berlangsung. Pada konteks sistem berbasis web, Wandri et al., (2025) menunjukkan bahwa pendekatan Agile Scrum cocok diterapkan pada pengembangan sistem berbasis web karena memungkinkan tim menyesuaikan arah pengembangan seiring munculnya kebutuhan baru dari pengguna, tanpa mengorbankan efisiensi proses. Stray et al., (2022) menghimpun praktik dan hasil penelitian mutakhir dari XP 2022 di bidang Agile dan Extreme Programming yang menjadi fondasi konseptual bagi pendekatan Scrum modern dalam pengembangan perangkat lunak.

Tonggiroh et al., (2024) membahas konsep dasar dan tahapan pengembangan perangkat lunak secara komprehensif, termasuk metodologi modern seperti Agile Scrum dan pendekatan berorientasi objek yang relevan dengan penelitian ini.

Dalam penelitian ini, metode *Agile Scrum* digunakan karena sistem yang dirancang dan dibangun memiliki beberapa fitur yang saling terintegrasi, seperti konfigurasi visual produk, kalkulasi harga *real-time*, *checkout*, pelacakan pesanan, pengelolaan data admin, verifikasi pembayaran, dan pembuatan SPK digital. Melalui pendekatan sprint, setiap fitur dapat dikembangkan secara bertahap, diuji, dievaluasi, dan diperbaiki sebelum digabungkan menjadi sistem yang utuh. Dengan demikian, *Agile Scrum* sesuai digunakan pada penelitian rancang bangun visual *product configurator* pakaian berbasis web karena mendukung proses pengembangan yang fleksibel, terstruktur, dan berorientasi pada hasil sistem yang dapat digunakan.

12. Smoke Testing

Smoke testing merupakan teknik pengujian awal yang digunakan untuk memastikan bahwa fungsi-fungsi utama pada sistem berjalan dengan baik. Pengujian ini dilakukan sebelum pengujian yang lebih mendalam, dengan tujuan mengetahui apakah sistem cukup stabil untuk digunakan atau diuji lebih lanjut. *Smoke testing* berfokus pada fungsi dasar atau alur utama, bukan pada seluruh kemungkinan skenario secara mendalam.

Anardani et al., (2024) menjelaskan bahwa *smoke testing* merupakan pengujian dasar yang menguji fungsi-fungsi utama sistem informasi untuk memastikan bahwa build cukup stabil dan siap untuk pengujian lebih lanjut, sehingga masalah pada alur utama dapat terdeteksi sejak awal sebelum dilakukan pengujian yang lebih detail.

Dalam penelitian ini, *smoke testing* digunakan untuk menguji fitur utama sistem visual *product configurator* pakaian berbasis web. Pada sisi pelanggan, fitur yang diuji meliputi *registrasi*, login, melihat jenis pakaian, konfigurasi visual pakaian, pemilihan komponen, pemilihan warna, pemilihan ukuran, jumlah pesanan, kalkulasi harga otomatis, unggah desain, *checkout*, riwayat pesanan, detail pesanan, dan pelacakan pesanan. Pada sisi admin, fitur yang diuji meliputi login admin, pengelolaan data jenis pakaian, pengelolaan aset produk, pengelolaan opsi kustomisasi, pengelolaan aturan harga, pengelolaan pesanan, verifikasi pembayaran, pembaruan status pesanan, dan pembuatan SPK digital dalam format PDF.

Smoke testing relevan dengan penelitian ini karena sistem yang dikembangkan memiliki alur utama yang harus berjalan tanpa kegagalan. Apabila fitur konfigurasi, *checkout*, *tracking*, verifikasi pembayaran, atau pembuatan SPK tidak berjalan, maka sistem belum dapat mendukung proses pemesanan pakaian kustom secara optimal. Oleh karena itu, *smoke testing* digunakan untuk memastikan bahwa fitur utama pada sisi pelanggan dan admin telah berjalan sesuai kebutuhan.

13. Sintesis Kajian Teoritis

Berdasarkan uraian tersebut, kajian teoritis penelitian ini dibangun dari keterkaitan antara *mass customization*, *product configurator*, visual *product configurator*, teknik *2D layering*, *state management*, *Object-Oriented Design*, *UML*, *Agile Scrum*, dan *smoke testing*. *Mass customization* memberikan dasar bahwa pelanggan perlu dilayani secara personal tanpa menghilangkan efisiensi. *Product configurator* menjadi mekanisme penerjemahan kebutuhan pelanggan ke spesifikasi produk yang valid. Visual *configurator* pada *apparel* menegaskan pentingnya antarmuka yang interaktif dan mudah dipahami.

Visual *product configurator* memperkuat proses konfigurasi karena pelanggan dapat melihat representasi visual dari produk yang dipilih. Teknik *2D layering* digunakan untuk menampilkan visualisasi pakaian secara ringan dan interaktif melalui penyusunan beberapa layer visual. *State management* digunakan untuk menjaga konsistensi data ketika pelanggan melakukan perubahan pilihan pada *configurator*. OOD dan UML digunakan untuk merancang struktur dan perilaku sistem sebelum implementasi. *Agile Scrum* digunakan sebagai metode pengembangan sistem secara bertahap, sedangkan *smoke testing* digunakan untuk memastikan fitur utama sistem berjalan sesuai kebutuhan.

Dengan demikian, seluruh teori tersebut saling mendukung dalam penelitian rancang bangun visual *product configurator* pakaian berbasis web. Teori-teori tersebut menjadi landasan dalam menganalisis masalah,

merancang sistem, mengembangkan fitur, dan menguji sistem yang dibangun untuk mendukung proses pemesanan pakaian kustom pada Nata Bordir.

B. Kajian Empiris

Kajian empiris penelitian ini disusun dari penelitian-penelitian terdahulu yang relevan pada rentang 2022–2026. Secara umum, penelitian terdahulu pada ranah *apparel* digital dan *product configurator* bergerak pada empat fokus besar, yaitu desain interaksi *configurator*, otomasi desain dan produksi pakaian, *virtual fitting* dan simulasi digital, serta elicitation kebutuhan pelanggan pada lingkungan *co-design*. Kajian ini digunakan untuk menunjukkan posisi penelitian yang dilakukan sekaligus mengidentifikasi celah penelitian yang masih terbuka.

Yi et al., (2022) menunjukkan bahwa desain interaksi pada *product configurator* berpengaruh terhadap pengalaman personalisasi konsumen. Temuan ini penting karena memperlihatkan bahwa keberhasilan *configurator* tidak semata ditentukan oleh banyaknya fitur, tetapi juga oleh bagaimana alur interaksi dirancang agar nyaman dan mudah digunakan. Relevansinya dengan penelitian ini terletak pada kebutuhan untuk membangun antarmuka *configurator* pakaian yang interaktif, jelas, dan tidak membingungkan bagi pengguna. Namun, penelitian tersebut belum membahas integrasi *configurator* dengan *checkout*, pelacakan pesanan, dan keluaran operasional seperti SPK digital.

Jin et al., (2023) menawarkan alternatif otomatisasi pembuatan pola pakaian dengan menggabungkan parameter desain dan skrip *Python*, sebagai upaya menekan pemborosan waktu, tenaga, dan material yang biasa terjadi saat pakaian dipersonalisasi secara manual. Hasilnya menunjukkan bahwa sistem tersebut mampu menghasilkan pola dengan akurasi dan kenyamanan yang tinggi serta mendukung proses produksi secara lebih efisien. Penelitian ini relevan karena menunjukkan bahwa digitalisasi pada industri pakaian dapat mempercepat respons terhadap kebutuhan personalisasi. Akan tetapi, fokus penelitian Jin et al. berada pada otomasi *pattern generation* untuk sisi produksi, bukan pada visual *configurator* berbasis web yang langsung digunakan pelanggan saat melakukan pemesanan.

Wang et al., (2023) mengusulkan pendekatan desain pakaian kustom berbasis simulasi digital *3D* dan *machine learning-supported human-product interactions*. Penelitian ini menunjukkan bahwa desain pakaian dapat dipersonalisasi melalui estimasi dimensi tubuh, pembentukan solusi desain, dan evaluasi hubungan antara profil konsumen dengan profil pakaian. Penelitian tersebut relevan karena sama-sama menempatkan pengguna sebagai pusat personalisasi produk. Meskipun demikian, pendekatan yang digunakan lebih kompleks karena bertumpu pada simulasi *3D* dan *machine learning*, sedangkan penelitian ini diarahkan pada solusi yang lebih ringan, terjangkau, dan operasional untuk konteks layanan konveksi lokal berbasis web.

Zhang et al., (2024) memakai pendekatan *Kano-QFD* untuk menerjemahkan kebutuhan pengguna aplikasi virtual *fitting* menjadi kebutuhan desain konkret, dengan prioritas tinggi pada *visual*, *emotional*, dan *interaction design* (terutama *personal image customization*, *realistic rendering*, dan *streamlined interaction*). Temuan ini relevan karena menegaskan bahwa pengguna digital *fashion* memprioritaskan pengalaman visual yang jelas dan mudah digunakan, tetapi fokusnya tetap pada virtual *fitting* dan alokasi sumber daya desain, bukan *configurator* web ringan yang terhubung dengan kalkulasi harga, *checkout*, pelacakan pesanan, dan SPK digital.

Dalam konteks UMKM di Indonesia, Rusmara & Ma'sum (2025) mengembangkan sistem kasir berbasis *web* dengan *framework* Laravel yang mampu mempercepat pencatatan transaksi, mempercepat proses pembayaran, serta menghasilkan laporan keuangan secara otomatis, sementara Setiawan, (2026) mengembangkan sistem informasi pemesanan online berbasis Laravel untuk UMKM Dapur Puni yang mampu mempercepat proses pemesanan, meningkatkan efisiensi, serta memperluas akses pelanggan. Kedua penelitian tersebut menegaskan bahwa Laravel merupakan *framework* yang tepat untuk digitalisasi UMKM lokal. Meski demikian, keduanya masih terbatas pada sistem pemesanan tekstual dan belum menyediakan *configurator* visual maupun keluaran operasional produksi yang menjadi fokus penelitian ini.

Lu, (2026) mengembangkan *intelligent clothing personalized customization system* berbasis *deep learning* dengan arsitektur *microservices* dan menunjukkan bahwa personalisasi pakaian secara online dapat berjalan lebih akurat dan responsif ketika didukung otomasi cerdas untuk menerjemahkan preferensi pengguna menjadi spesifikasi produk. Temuan tersebut relevan dengan penelitian ini karena menegaskan pentingnya sistem yang mampu memandu pengguna non-ahli dalam menyampaikan kebutuhan produk secara terstruktur. Namun, penelitian tersebut berfokus pada mesin personalisasi berbasis *deep learning* dan belum mencakup sistem *end-to-end* dari konfigurasi visual, kalkulasi harga, *checkout*, pelacakan pesanan, hingga keluaran administratif dan produksi.

Studi terdahulu telah berkontribusi pada aspek *interaction design*, *virtual fitting*, *automated garment design*, dan *customer requirement elicitation*, tetapi masih terbatas yang membangun *visual product configurator* pakaian berbasis web yang ringan dengan teknik *2D layering* dan *state management*, sekaligus mengintegrasikan kalkulasi harga, *checkout*, pelacakan pesanan, dan SPK digital. Penelitian ini mengisi celah tersebut dengan menghubungkan visualisasi konfigurasi produk langsung ke alur administrasi dan produksi pada Nata Bordir.

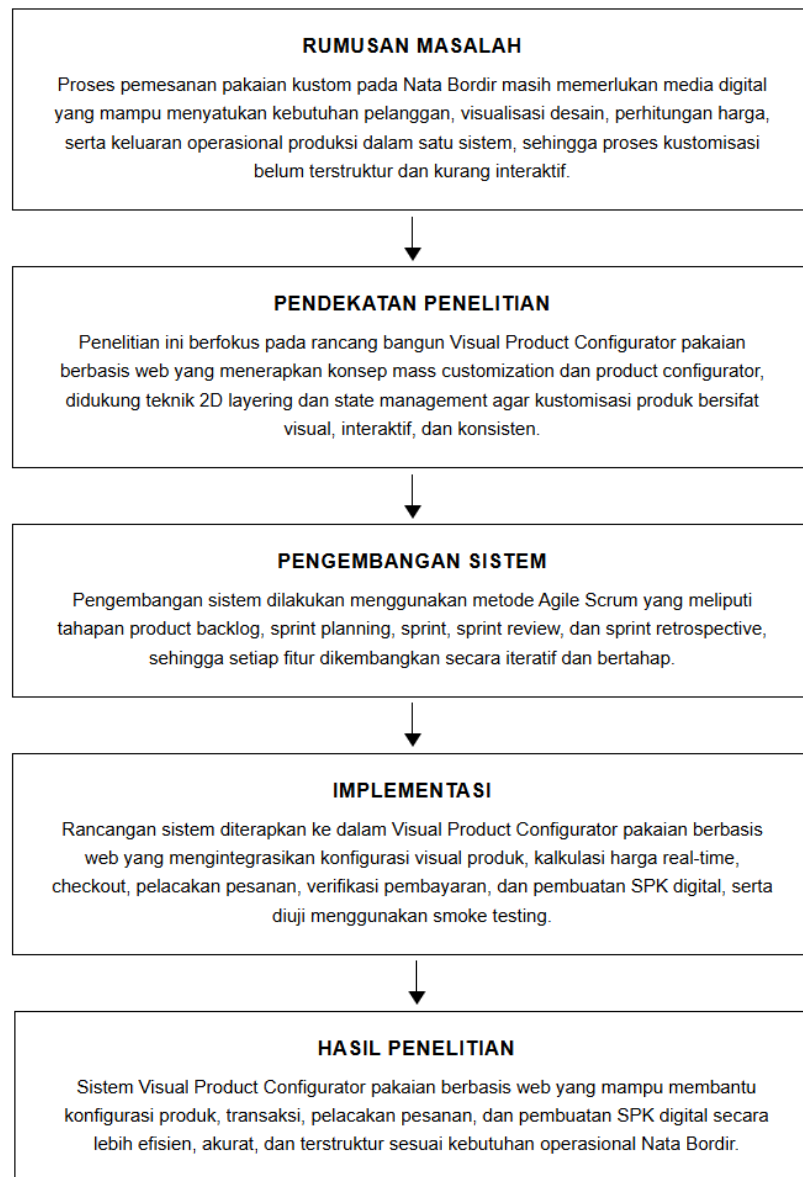
C. Kerangka Berpikir

Penelitian ini berangkat dari kebutuhan Nata Bordir akan satu sistem berbasis web yang menyatukan kebutuhan pelanggan, visualisasi desain, perhitungan harga, dan keluaran produksi. Kebutuhan tersebut sejalan

dengan konsep *mass customization* dan *product configurator*, yaitu pendekatan yang menyesuaikan produk dengan preferensi individual pelanggan tanpa mengorbankan efisiensi sekaligus menerjemahkan kebutuhan menjadi konfigurasi produk yang valid (Bouchard et al., 2023; Uduwela et al., 2023).

Secara teknis, sistem menggunakan *2D layering* dan *state management* agar visualisasi produk tetap ringan, interaktif, dan konsisten saat pelanggan mengubah pilihan konfigurasi (Balla dan Gede, 2025; Veeri, 2024), sementara penelitian terdahulu belum mengintegrasikan konfigurasi visual, kalkulasi harga, *checkout*, pelacakan pesanan, dan pembuatan SPK digital dalam satu sistem web untuk usaha konveksi lokal.

Untuk menjawab kebutuhan tersebut, penelitian ini merancang dan membangun *visual product configurator* pakaian berbasis web dengan metode *Agile Scrum* yang iteratif melalui tahapan *product backlog*, *sprint planning*, *sprint*, *sprint review*, dan *sprint retrospective*. Alur kerangka berpikirnya dapat dilihat pada Gambar 2.1.



Gambar 2. 1 Kerangka Berpikir