

BAB II

KAJIAN PUSTAKA

A. Kajian Teoritis

1. Manajemen Inventaris dan Stok Mati (*Deadstock*)

Manajemen inventaris merupakan proses krusial dalam pengendalian stok barang yang mencakup kegiatan pencatatan, pemantauan, hingga pengendalian persediaan untuk memastikan ketersediaan produk sesuai kebutuhan tanpa mengabaikan efisiensi biaya. (Hanani et al., 2025) menjelaskan bahwa banyak UMKM yang masih mengandalkan manajemen inventaris manual melalui buku catatan atau *spreadsheet* sederhana, yang sangat rentan terhadap *human error*, ketidakakuratan data, hingga risiko berkurangnya keuntungan finansial. Pengelolaan manual ini seringkali tidak mampu memberikan visibilitas yang cukup terhadap pergerakan barang, sehingga operasional toko menjadi kurang efisien dan sulit untuk dimonitor secara tepat waktu.

Dalam manajemen persediaan, kegiatan *stock opname* menjadi instrumen vital untuk mencocokkan data sistem dengan jumlah fisik aktual di gudang. (C. A. Putri & Handoko, 2024) menegaskan bahwa proses *stock opname* yang dilakukan secara manual sering kali mengalami kendala berupa data yang sering tidak valid, memakan waktu lama, serta sering ada kesalahan manusia. Jika sistem pengelolaan masih manual, perbedaan data antara jumlah di sistem dengan fisik sering kali luput dari pengawasan, yang pada akhirnya

dapat mengakibatkan kerugian finansial yang signifikan bagi usaha ritel karena fungsi pengendalian internal tidak berjalan dengan baik.

Dampak yang paling terlihat dari pengelolaan yang tidak optimal adalah munculnya *deadstock* atau stok mati, yaitu barang yang mengendap di gudang dalam jangka waktu lama tanpa perputaran penjualan yang jelas. Basyarahil (Fahmi et al., 2026) menyebutkan bahwa pengambilan keputusan pengadaan barang yang hanya bergantung pada opini atau pengalaman pribadi tanpa analisis data yang tepat dapat memicu penumpukan stok yang tidak laku. Penumpukan *deadstock* ini merugikan bisnis karena modal usaha tertahan pada barang yang tidak produktif, sekaligus memperbesar risiko kerugian akibat kerusakan fisik atau produk yang kadaluarsa sebelum terjual.

Digitalisasi manajemen inventaris menjadi langkah strategis untuk mengurangi ketergantungan pada pencatatan manual, meningkatkan akurasi data stok, serta meminimalisir risiko penumpukan barang yang tidak bergerak. Pemanfaatan sistem informasi yang terkomputerisasi memungkinkan pengelolaan data secara otomatis dan akurat, sehingga pemilik usaha dapat memantau kesehatan stok secara *real-time*. Dengan dukungan teknologi informasi, proses *data mining* dapat diterapkan untuk mengolah data historis guna memberikan rekomendasi keputusan yang objektif, yang pada akhirnya membantu meminimalkan risiko kerugian modal akibat barang yang tidak diminati pasar.

2. Definisi *Data Mining*

Data mining atau yang sering disebut sebagai *Knowledge Discovery in Databases* (KDD) merupakan rangkaian proses ekstraksi untuk menemukan pengetahuan, pola, atau informasi tersembunyi di dalam sekumpulan data yang sangat besar. (Handayani, 2022) menjelaskan bahwa *data mining* berfungsi sebagai metode untuk mengubah data mentah menjadi informasi yang bernilai strategis bagi pengambil keputusan. Dengan memanfaatkan teknik ini, data transaksi yang selama ini hanya tersimpan sebagai arsip dapat digali untuk menghasilkan pola perilaku bisnis yang lebih komprehensif bagi pemilik usaha.

Proses *data mining* melibatkan tahapan sistematis yang dimulai dari seleksi data, *preprocessing*, transformasi, penerapan algoritma, hingga interpretasi pola. (Dewi & Rahayu, 2022) menekankan bahwa tahapan *preprocessing* atau pembersihan data menjadi aspek paling krusial, karena kualitas data yang diolah akan menentukan akurasi hasil analisis akhir. **Proses** pembersihan data meliputi menghapus data duplikat, mengecek data yang tidak konsisten, dan memperbaiki data yang keliru. Tanpa melalui proses KDD yang runtut dan benar, hasil penambangan data tidak akan memberikan wawasan (*insight*) yang valid untuk mendukung kebijakan strategis perusahaan yang berkelanjutan.

Teknik *data mining* secara garis besar diklasifikasikan menjadi kategori deskriptif dan prediktif, di mana penggunaannya sangat bergantung pada kebutuhan spesifik bisnis. (Firmansyah & Nurdiawan, 2023) dalam penelitiannya menunjukkan bahwa penerapan teknik *data mining* dengan

algoritma asosiasi seperti *Frequent Pattern-Growth* (FP-Growth) mampu menghasilkan pola pembelian produk yang signifikan, sehingga perusahaan dapat menyusun strategi pemasaran yang lebih efektif seperti penyusunan paket produk (*bundling*) yang tepat sasaran. Penerapan algoritma ini memungkinkan pemilik usaha untuk memahami perilaku konsumen dan meningkatkan efisiensi penjualan secara objektif.

Kesimpulannya, *data mining* adalah instrumen strategis untuk beralih dari pengambilan keputusan berbasis intuisi menjadi keputusan yang berbasis data. Bagi Toko Plastik T'Four, implementasi teknik ini melalui algoritma K-Means dan FP-Growth sangat relevan untuk mengeksploitasi data transaksi, sehingga masalah penumpukan *deadstock* dapat dideteksi secara dini sekaligus mengoptimalkan strategi pemasaran guna mencapai efisiensi operasional yang maksimal.

3. Algoritma K-Means Clustering

K-Means merupakan algoritma *clustering* berbasis partisi yang sangat populer dalam dunia *data mining* karena kemampuannya dalam mengelompokkan data ke dalam beberapa kelompok (*cluster*) berdasarkan kemiripan karakteristiknya. (Asmana et al., 2022) menjelaskan bahwa algoritma ini adalah metode yang paling sederhana namun banyak digunakan dalam tugas *data mining* karena kemudahannya dalam implementasi serta kecepatannya dalam memproses data.

Mekanisme kerja *K-Means* melibatkan proses partisi data ke dalam sejumlah *cluster* yang telah ditentukan di awal dengan meminimalkan jarak

rata-rata antara setiap titik data dengan pusat *cluster* (*centroid*). (Pujiono et al., 2024) menyebutkan bahwa algoritma ini bekerja secara iteratif untuk mengoptimalkan pengelompokan dengan memastikan setiap titik data berada pada jarak minimum dari pusat *cluster* yang bersangkutan. Secara matematis, jarak ini dihitung menggunakan rumus *Euclidean Distance*:

$$d(x, y) = \sqrt{\sum_{i=1}^n (x_i - y_i)^2}$$

Keterangan: $d(x,y)$ adalah jarak antara data x dan pusat cluster y , x_i adalah atribut data, y_i adalah nilai centroid, dan n adalah jumlah dimensi data.

(Yani et al., 2023) menambahkan bahwa posisi pusat *cluster* akan diberi tanda sehingga dapat mengelompokkan seluruh data komponen-komponen sebelumnya yang memiliki jarak terdekat dengan pusat *cluster*.

Penerapan *K-Means* sangat strategis dalam operasional bisnis untuk mengatasi kesulitan dalam pengelolaan stok barang yang sering kali tidak efisien jika dilakukan secara manual. Dalam konteks pengelolaan inventaris, algoritma ini memungkinkan pemilik usaha untuk mengategorikan produk ke dalam kelompok laris, sedang, atau kurang laris sehingga langkah promosi dan manajemen stok dapat dilakukan lebih akurat. Dengan bantuan teknologi *data mining* ini, keputusan manajerial terkait ketersediaan barang dapat diambil dengan lebih cepat dan objektif, guna meminimalkan penumpukan stok yang merugikan perusahaan.

K-Means merupakan solusi efektif dalam melakukan segmentasi data transaksi untuk mendukung efisiensi manajemen inventaris. Dengan

mengoptimalkan pembagian kelompok, algoritma ini memberikan hasil pengelompokan yang presisi bagi pelaku usaha. Hal ini menjadikan *K-Means* sebagai instrumen pendukung keputusan yang sangat membantu dalam menentukan strategi pengelolaan barang secara akurat.

4. Algoritma FP-Growth (*Frequent Pattern Growth*)

Algoritma *FP-Growth* (*Frequent Pattern Growth*) merupakan teknik penambangan data yang dioptimalkan untuk mengekstraksi aturan asosiasi, yakni menemukan pola produk yang paling sering dibeli secara bersamaan oleh konsumen. Menurut (Rustam et al., 2024), pertumbuhan data penjualan yang dalam pembuatan analisisnya masih manual membuat sulit untuk mengetahui pola yang tersembunyi. Oleh karena itu, penggunaan algoritma ini menjadi solusi krusial untuk mengidentifikasi catatan yang sering muncul dalam dataset besar secara otomatis, yang mana hal ini sangat sulit dilakukan jika sistem pengelolaan data masih dilakukan secara konvensional atau manual.

Penerapan *FP-Growth* berfokus pada efisiensi pemrosesan data transaksi dengan menentukan nilai ambang batas tertentu, seperti *minimum support* dan *minimum confidence*, untuk menghasilkan aturan asosiasi yang signifikan. Perhitungan parameter tersebut dilakukan dengan menggunakan rumus berikut:

$$\text{Support}(A) = \frac{\text{Jumlah Transaksi mengandung } A}{\text{Total Transaksi}}$$

$$\text{Confidence}(A \rightarrow B) = \frac{\text{Jumlah Transaksi mengandung } A \text{ dan } B}{\text{Jumlah Transaksi mengandung } A}$$

Nilai *Support* digunakan untuk mengetahui seberapa sering kombinasi produk muncul dalam dataset, sedangkan *Confidence* digunakan untuk mengukur seberapa kuat hubungan antar produk tersebut. Selain itu, untuk menguji validitas dan kekuatan aturan asosiasi, digunakan parameter *Lift Ratio* dengan persamaan berikut:

$$\text{Lift}(A \rightarrow B) = \frac{\text{Confidence}(A \rightarrow B)}{\text{Support}(B)}$$

Lift Ratio digunakan untuk mengukur kekuatan korelasi antara produk A dan produk B dibandingkan jika keduanya dibeli secara independen. Jika nilai *Lift Ratio* > 1 , maka terdapat hubungan positif yang kuat antara kedua produk tersebut, yang mengindikasikan bahwa produk A dan B sangat layak untuk dipaketkan (*bundling*). Sebaliknya, jika nilai *Lift Ratio* ≤ 1 , maka hubungan antar produk dianggap tidak signifikan atau bahkan berlawanan.

(Nurarofah et al., 2023) dalam penelitiannya menunjukkan bahwa penerapan algoritma ini sangat efektif untuk mengatasi permasalahan tata letak produk yang tidak sesuai, sehingga memudahkan pelanggan selama proses pembelian. Algoritma ini memungkinkan sistem untuk memilah mana jenis produk yang kerap dibeli bersamaan oleh konsumen, sehingga memberikan varian hubungan produk yang bermakna bagi pengelola toko dalam mengelola katalog produk.

Tujuan utama dari implementasi *FP-Growth* adalah untuk mengoptimalkan strategi penjualan melalui identifikasi pola pembelian yang diminati pelanggan secara spesifik. Sejalan dengan hal tersebut, (Asih & Martanto, 2024) menekankan bahwa algoritma ini berperan besar dalam

menentukan produk yang paling diminati dan melihat banyaknya itemset yang memiliki jumlah paling besar dalam sebuah dataset, seperti kombinasi item belanja yang memiliki tingkat kepercayaan (*confidence*) tinggi. Hasil dari penambahan data ini memberikan informasi berharga bagi pemilik usaha untuk menyusun paket *bundling* atau promosi produk, yang pada akhirnya meningkatkan efektivitas penjualan dan meminimalkan stok barang yang tidak diminati pasar.

Algoritma *FP-Growth* adalah metode yang andal untuk mengubah data transaksi mentah menjadi strategi bisnis yang bernilai melalui *Market Basket Analysis*. Dengan kemampuan mengidentifikasi pola pembelian yang sering terjadi secara cepat dan akurat, metode ini menjadi aset penting bagi pelaku usaha untuk meningkatkan daya saing produk, mengoptimalkan tata letak barang, serta meningkatkan keuntungan penjualan melalui strategi *bundling* yang tepat sasaran.

5. Pemrograman Python

Python adalah bahasa pemrograman *high-level* yang sangat fleksibel dan banyak digunakan karena kemudahannya dalam memproses data serta dukungan *library* yang sangat luas. (Surbakti et al., 2024) menjelaskan bahwa penggunaan Python sangat efektif dalam memodelkan fungsi matematika dan melakukan analisis komputasi kompleks, yang menjadikannya bahasa utama untuk berbagai kebutuhan sains dan pemrosesan data numerik. Kemampuan pustaka bawaan seperti *NumPy* dan *Matplotlib* memungkinkan pengembangan sistem untuk melakukan perhitungan data yang rumit dengan struktur kode yang

jauh lebih sederhana dibandingkan bahasa pemrograman konvensional lainnya, sehingga mempercepat fase pengembangan aplikasi.

Dalam implementasi sistem informasi, Python memiliki kapabilitas yang mumpuni dalam pengumpulan data mentah dari berbagai sumber internet melalui teknik *web scraping*. (Setiawan et al., 2024) menyatakan bahwa teknik *web scraping* dengan Python dapat dimanfaatkan untuk mengumpulkan data penjualan secara otomatis, yang kemudian diproses menjadi informasi strategis sebagai dasar pengambilan keputusan dalam bisnis. Hal ini menunjukkan bahwa Python tidak hanya berfungsi sebagai bahasa pengkodean, tetapi juga sebagai alat pendukung pengumpulan data (*data gathering*) yang krusial bagi sistem yang berbasis pada analisis data yang akurat.

Selain unggul dalam analisis data, Python juga terbukti handal dalam pengembangan aplikasi web dan e-commerce yang berorientasi pada optimalisasi kinerja penjualan. (W. D. Putri et al., 2025) menegaskan bahwa sistem berbasis Python mampu meningkatkan efisiensi operasional bisnis melalui rancang bangun aplikasi yang mengintegrasikan berbagai modul transaksi secara efisien. Dengan arsitektur yang fleksibel dan dukungan pustaka yang luas, Python memberikan landasan bagi pengembang untuk membangun sistem yang tidak hanya berfokus pada fungsionalitas teknis, tetapi juga pada optimalisasi hasil penjualan dan pengelolaan data yang lebih terorganisir.

Python adalah fondasi utama dalam pengembangan sistem berbasis data karena fleksibilitasnya dalam memproses angka, kemampuannya dalam mengotomatisasi pengambilan data, serta efektivitasnya dalam membangun

arsitektur sistem informasi yang modern. Keandalan pustaka *library* Python menjadikan bahasa ini sebagai pilihan utama untuk mengintegrasikan logika analisis data dengan antarmuka aplikasi web yang responsif guna mendukung pengambilan keputusan yang lebih baik dalam sebuah bisnis.

6. Framework Streamlit

Streamlit adalah *framework* berbasis Python (*open-source*) yang dirancang khusus untuk mempermudah pengembang dalam membangun program berbasis web interaktif di bidang *data science* dan *machine learning*. (Azhar et al., 2024) menyatakan bahwa *Streamlit* mampu menerjemahkan model analitik dan algoritma yang rumit menjadi aplikasi visual yang fungsional dalam waktu yang singkat. Kecepatan *render* komponen ini memangkas waktu pengembangan antarmuka (*UI/UX*) secara drastis, sehingga pengembang bisa mengalihkan fokus sepenuhnya pada logika algoritma *backend* tanpa harus terbebani oleh kompleksitas pemrograman *front-end* tradisional seperti HTML atau *JavaScript* yang memakan waktu lama.

Keunggulan utama dari *framework* ini terletak pada sistem interaktivitas bawaannya yang sangat ramah terhadap pengguna awam. (Devalio et al., 2025) menjelaskan bahwa *Streamlit* mendukung penyajian informasi yang memanfaatkan komponen dinamis serta kemudahan navigasi bagi *end-user* dalam mengakses sistem berbasis web secara interaktif. Ketika pengguna memasukkan input data atau menekan tombol model algoritma pada layar, *Streamlit* secara otomatis memperbarui visualisasi grafik tanpa perlu

memuat ulang (*reload*) keseluruhan halaman, yang secara signifikan meningkatkan kecepatan responsivitas sistem terhadap permintaan pengguna.

Implementasi *Streamlit* dalam riset data sangat strategis untuk menjembatani kerumitan algoritma Python dengan kebutuhan visualisasi pengguna akhir. (Syafarina, 2023) membuktikan dalam penelitiannya bahwa penggunaan *Streamlit* berhasil menciptakan aplikasi berbasis kecerdasan buatan (*Artificial Intelligence*) yang mampu melakukan prediksi data dengan antarmuka yang intuitif dan mudah dioperasikan oleh pengguna. Hal ini menjadikannya solusi visualisasi yang paling ideal untuk berbagai sektor, karena aplikasi yang dibangun tidak hanya mementingkan akurasi hasil komputasi, tetapi juga memperhatikan aspek kemudahan akses bagi pengguna dalam mengambil keputusan berbasis data yang cepat dan akurat.

Streamlit merupakan fondasi yang efisien dalam pengembangan sistem informasi modern karena fleksibilitasnya dalam memproses logika Python menjadi antarmuka web interaktif secara instan. Dengan menggabungkan kecepatan pengembangan, responsivitas antarmuka, dan kemudahan bagi pengguna awam, *Streamlit* terbukti menjadi instrumen visualisasi data yang andal untuk mendukung pengambilan keputusan yang berbasis data di berbagai bidang.

7. Basis Data SQLite

SQLite merupakan sistem manajemen basis data relasional (*Relational Database Management System*) yang dirancang dengan karakteristik *serverless* dan sangat ringan. Menurut (Rachman et al., 2025), *SQLite* berfungsi sebagai

basis data lokal yang umum digunakan dalam pengembangan aplikasi supaya dapat dijalankan dengan performa yang ringan dan simpel dengan adanya ACID membuat SQLite handal untuk menjaga integritas data menjadikannya sangat praktis bagi pengembang perangkat lunak, terutama dalam aplikasi yang membutuhkan akses data cepat dan penyimpanan lokal yang stabil untuk mendukung fungsionalitas aplikasi.

Penggunaan sistem basis data dalam operasional usaha kecil memberikan dampak yang signifikan terhadap efisiensi dan akurasi pengelolaan informasi penjualan. (Azizah & Razaqi, 2026) menyatakan bahwa pengelolaan data meningkat hingga 90% dan tingkat kesalahan dalam perekaman manual bisa menurun. Dengan transisi dari sistem pencatatan manual ke sistem basis data, perusahaan dapat menjamin keamanan data transaksi serta mempercepat layanan operasional yang sebelumnya terhambat oleh proses administrasi yang tidak terstruktur.

Dari sisi performa, *SQLite* telah teruji keandalannya dalam menangani berbagai operasi pengolahan data yang mendukung kelancaran proses bisnis. (Nugraha & Susetyo, 2023) dalam penelitiannya menegaskan bahwa pemilihan basis data yang tepat sangat penting agar pengolahan data tidak melambat seiring bertambahnya kompleksitas informasi, sehingga proses bisnis tetap berjalan optimal. Karakteristik ini membuat *SQLite* menjadi pilihan yang sangat rasional untuk diintegrasikan dalam aplikasi web berbasis *Streamlit*, karena mampu menjaga integritas data transaksi penjualan yang masuk secara lokal tanpa membebani memori sistem.

SQLite merupakan solusi basis data yang efisien untuk aplikasi berbasis web skala kecil hingga menengah karena sifatnya yang *serverless* dan ringan. Dengan mengintegrasikan sistem basis data ke dalam aplikasi, pengelolaan data transaksi dapat dilakukan dengan tingkat akurasi yang lebih tinggi, meminimalkan risiko kesalahan manusia, serta mempercepat akses informasi yang dibutuhkan untuk pengambilan keputusan bisnis yang tepat.

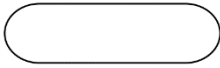
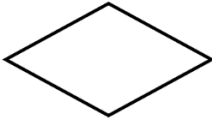
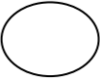
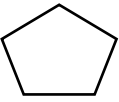
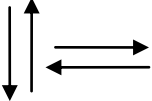
8. *Flowchart* (Diagram Alir)

Flowchart atau yang sering disebut sebagai diagram alir, adalah perangkat visual yang menggunakan simbol-simbol standar untuk menggambarkan urutan langkah atau prosedur kerja dari sebuah program secara sistematis. Menurut (Trisnanto, 2024), *flowmap* digambarkan melalui dimensi 2D secara grafik dari langkah-langkah dan urutan prosedur sebuah program yang berfungsi untuk membantu analisis dan programmer dalam memecahkan masalah ke dalam segmen-segmen yang lebih kecil. Dengan adanya diagram ini, kompleksitas proses bisnis dapat disederhanakan menjadi representasi visual yang memudahkan pemahaman terhadap alur data yang masuk maupun keluar dari sistem.

Sebagai sebuah cetak biru (*blueprint*), *flowchart* menjadi elemen fundamental sebelum pengembang mulai menulis kode program. (Hasanudin, 2022) menegaskan bahwa *Flowchart* sangat penting untuk memetakan alur kerja secara terstruktur agar sistem yang dibangun dapat beroperasi dengan efisien sesuai dengan kebutuhan pengguna. Agar diagram tersebut dapat dibaca

secara universal, penggunaan simbol-simbol standar sangat diperlukan. Berikut adalah tabel simbol yang umum digunakan dalam perancangan *flowchart*.

Tabel 2.1 Simbol *Flowchart*

No	Simbol	Nama Simbol	Fungsi Simbol
1		Terminal	Awal atau akhir dari suatu program atau prosedur.
2		Input/Output	Proses memasukkan data atau menampilkan hasil data.
3		Process	Proses pengolahan data atau instruksi program.
4		Decision	Pengambilan keputusan (kondisi benar/salah atau ya/tidak).
5		Connector	Menghubungkan alur proses dalam satu halaman yang sama.
6		Predefined Process	Simbol untuk menyediakan tempat-tempat pengolahan data dalam <i>storage</i> .
7		Flow	Menyatakan arah atau jalannya arus suatu proses.

Sumber: (Kurniawan et al., 2026)

Selain mempermudah pemahaman alur kerja, *flowchart* juga berfungsi sebagai standar dokumentasi sistem yang vital. (Kurniawan et al., 2026) menjelaskan bahwa dalam pengembangan sistem informasi manajemen, perancangan yang baik melalui berbagai diagram, termasuk *flowchart*, sangat krusial untuk memastikan integrasi antara pelaku, proses, dan data berjalan dengan transparan. Dengan pemetaan yang jelas melalui simbol-simbol di atas, pengembang dapat melacak setiap perubahan data secara presisi, yang pada akhirnya mempermudah proses pemeliharaan sistem di masa mendatang.

Flowchart merupakan instrumen perancangan sistem yang sangat krusial dalam pengembangan perangkat lunak untuk memetakan logika proses menjadi alur yang terstruktur. Melalui penggunaan simbol standar yang baku, pengembang dapat menyederhanakan masalah yang kompleks menjadi langkah-langkah yang mudah diimplementasikan, sekaligus menyediakan dokumentasi standar yang memudahkan integrasi data dalam sistem informasi yang dibangun.

9. *Unified Modeling Language* (UML)

Unified Modeling Language (UML) merupakan bahasa pemodelan visual yang menjadi standar dalam merancang dan mendokumentasikan perangkat lunak. (Wayahdi & Ruziq, 2023) menyatakan bahwa pengembangan sistem yang baik memerlukan analisis mendalam dan dokumentasi baku, di mana UML hadir untuk memenuhi standar tersebut. Melalui UML, kita bisa memetakan alur logika aplikasi sebelum mulai menulis kode untuk meminimalkan kesalahan arsitektur sejak awal.

UML memecah kompleksitas sistem menjadi beberapa jenis diagram berdasarkan fungsinya. (HUDA, 2025) menjelaskan bahwa diagram *Use Case* berfungsi mendefinisikan interaksi pengguna dengan sistem, sementara diagram *Activity* memvisualisasikan alur kerja atau proses bisnis yang berjalan. Pemisahan peran diagram ini memudahkan tim pengembang membaca alur sistem secara spesifik tanpa kehilangan pemahaman terhadap arsitektur secara keseluruhan.

Implementasi UML memberikan kerangka yang terstruktur dalam perancangan aplikasi. (Kurniawan et al., 2026) menegaskan bahwa UML adalah pendekatan pemodelan visual yang efektif sebagai alat bantu merancang sistem berbasis objek. Pendekatan visual ini memastikan setiap batasan sistem dan hak akses pengguna terpetakan dengan akurat, sehingga menghasilkan cetak biru yang solid bagi *programmer* saat mengeksekusi kode program.

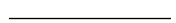
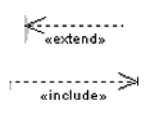
UML berfungsi sebagai jembatan komunikasi visual yang presisi antara analis sistem dan *programmer*. Pemodelan melalui diagram standar memastikan perangkat lunak yang dibangun selaras dengan batasan serta kebutuhan fungsional pengguna akhir.

a. Use Case Diagram

Use Case Diagram merupakan diagram yang memodelkan interaksi antara pengguna (*actor*) dengan sistem yang sedang dikembangkan. (Wayahdi & Ruziq, 2023) menjelaskan bahwa diagram ini mendefinisikan batasan sistem dengan sangat jelas, yaitu memisahkan mana entitas yang berada di luar sistem (aktor) dan mana fungsionalitas yang harus dikerjakan oleh sistem (kasus penggunaan).

Fokus utama diagram ini bukan pada bagaimana sistem bekerja secara teknis di balik layar, melainkan apa saja layanan yang bisa diakses oleh pengguna. (HUDA, 2025) menambahkan bahwa *Use Case Diagram* sangat efektif digunakan pada tahap analisis kebutuhan, karena mampu menerjemahkan ekspektasi pengguna ke dalam rancangan fungsional yang mudah dipahami oleh tim pengembang. Berikut adalah simbol-simbol standar yang digunakan.

Tabel 2.2 Simbol *Use Case Diagram*

No	Simbol	Nama	Fungsi
1		Aktor	Mewakili entitas eksternal (manusia atau sistem lain) yang berinteraksi langsung dengan sistem.
2		<i>Use case</i>	Menggambarkan fungsionalitas, layanan, atau pekerjaan utama yang disediakan oleh sistem.
3		Association	Garis penghubung yang menunjukkan interaksi langsung antara <i>actor</i> dengan <i>use case</i> .
4		<i>Include / Extend</i>	Garis putus-putus berpanah untuk menunjukkan ketergantungan spesifik (<i>include</i> untuk wajib, <i>extend</i> untuk opsional) antar <i>use case</i> .
5		<i>System Boundary</i>	Kotak pembatas yang memisahkan ruang lingkup sistem di bagian dalam dengan aktor di bagian luar.

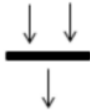
Sumber:(HUDA, 2025).

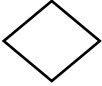
b. Activity Diagram

Activity Diagram berfungsi untuk memvisualisasikan alur kerja (*workflow*) atau proses bisnis dari sebuah sistem secara rinci. (Kurniawan et al., 2026) mendeskripsikan diagram ini sebagai pengembangan dari *Use Case* yang lebih fokus memodelkan urutan proses secara vertikal, mulai dari titik awal hingga proses tersebut berakhir.

Diagram ini memiliki kemiripan dengan *flowchart*, namun memiliki keunggulan karena mampu menggambarkan proses yang berjalan secara paralel atau bersamaan. Menurut (Wayahdi & Ruziq, 2023), *Activity Diagram* sangat berguna untuk mengontrol aliran tindakan tingkat tinggi maupun mendeskripsikan algoritma tingkat rendah sebelum implementasi kode dilakukan. Berikut adalah tabel simbol yang umum digunakan.

Tabel 2.3 Simbol Activity Diagram

Simbol	Nama	Fungsi
	<i>Initial Node</i>	Menandakan titik awal atau bagaimana suatu objek/alur sistem mulai dieksekusi.
	<i>Activity</i>	Mewakili status atau langkah eksekusi dari suatu aksi di dalam sistem.
	<i>Fork / Join</i>	Garis tebal yang menunjukkan satu aliran berubah menjadi beberapa aliran paralel (<i>fork</i>) atau sebaliknya (<i>join</i>).

	<i>Decision Node</i>	Titik pengambilan keputusan yang memecah alur berdasarkan kondisi tertentu (misalnya: berhasil/gagal).
	Activity Final Node	Menandakan titik akhir dari keseluruhan alur proses atau bagaimana objek dihancurkan.

Sumber: (Kurniawan et al., 2026)

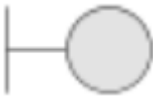
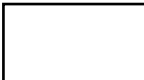
c. Sequence Diagram

Sequence diagram merupakan bagian dari UML yang memodelkan interaksi antar objek berdasarkan urutan waktu. Menurut (Kurniawan et al., 2026), diagram ini sangat penting untuk merinci aliran pesan (message) dan pertukaran data yang terjadi di dalam sistem sejak sebuah fungsi dipicu hingga menghasilkan output. Berbeda dengan activity diagram yang berfokus pada alur kerja, sequence diagram lebih menonjolkan aspek arsitektur teknis.

Diagram ini memetakan dengan jelas peran antarmuka (*front-end*), pengontrol logika (*back-end*), hingga basis data (*database*) saat mengeksekusi sebuah kasus penggunaan. Berikut adalah simbol-simbol standar yang digunakan dalam perancangan sequence diagram.

Tabel 2.4 Simbol *Sequence Diagram*

Simbol	Nama	Fungsi
	<i>Actor</i>	Mewakili pengguna atau sistem eksternal yang memicu jalannya sebuah proses atau interaksi.

	<i>Boundary</i>	Menggambarkan elemen antarmuka (<i>user interface</i>) tempat aktor berinteraksi, seperti halaman web atau <i>form</i> .
	<i>Control</i>	Mewakili mesin pemroses, logika algoritma, atau sistem <i>back-end</i> yang mengeksekusi perintah.
	<i>Entity / Database</i>	Mewakili media penyimpanan data persisten tempat sistem mengambil dan menyimpan informasi.
	<i>Lifeline</i>	Garis putus-putus vertikal yang menunjukkan keberadaan atau waktu aktif sebuah objek selama proses berlangsung.
	<i>Activation Box</i>	Kotak persegi panjang pada garis hidup yang menandakan objek tersebut sedang aktif mengeksekusi tugas.
	<i>Message</i>	Garis panah horizontal solid yang merepresentasikan pengiriman perintah atau pemanggilan metode ke objek lain.
	<i>Return Message</i>	Garis panah horizontal putus-putus yang menandakan pengembalian nilai atau hasil data ke pengirim awal.

Sumber:(Kurniawan et al., 2026)

d. *Class Diagram*

Class diagram merupakan pemodelan struktural dalam *Unified Modeling Language* (UML) yang memetakan rancangan statis sebuah sistem perangkat lunak. Secara visual, diagram ini mendeskripsikan susunan entitas secara rinci menggunakan komponen kotak tiga sekat. Sekat atas memuat nama kelas sebagai identitas entitas, sekat tengah berisi daftar atribut beserta tipe datanya, dan sekat

bawah menampung operasi atau fungsi logis yang dapat dieksekusi oleh kelas tersebut (HUDA, 2025).

Visualisasi ini berperan krusial sebagai cetak biru teknis bagi pengembang saat mengeksekusi penulisan kode program. Hubungan antar-kelas digambarkan lewat garis asosiasi utuh untuk menunjukkan interaksi statis. Garis ini dilengkapi dengan angka kardinalitas (*multiplicity*) seperti relasi tunggal (1) atau jamak (0..*) untuk memastikan arsitektur sistem dibangun secara presisi, sehingga mempermudah tahapan evaluasi fungsionalitas serta proses pemeliharaan aplikasi di masa mendatang (Wayahdi & Ruziq, 2023).

10. Pengujian *Black Box*

Pengujian *black box* merupakan metode pengujian perangkat lunak yang berfokus pada evaluasi fungsionalitas sistem tanpa perlu mengetahui struktur kode internalnya. (Devalio et al., 2025) menjelaskan bahwa pengujian *black box* dilakukan secara spesifik untuk memastikan seluruh fitur di dalam aplikasi dapat berfungsi dengan baik sesuai dengan perancangan awal. Metode ini sangat praktis karena penguji hanya perlu memberikan *input* tertentu ke dalam antarmuka sistem dan mengamati *output* yang dihasilkan, sehingga evaluasi murni berpusat pada perspektif pengguna akhir.

Dalam pengembangan sistem informasi, pengujian ini krusial untuk memvalidasi kelayakan operasional aplikasi sebelum benar-benar diimplementasikan. (Kurniawan et al., 2026) menunjukkan bahwa melalui pengujian *black box*, pengembang dapat memverifikasi kemampuan sistem dalam mengelola aktivitas pemrosesan data, seperti manipulasi data secara

presisi, menampilkan peringatan otomatis, hingga menghasilkan laporan yang akurat. Hal ini memastikan bahwa antarmuka pengguna dan fungsionalitas sistem telah terintegrasi dengan sempurna.

Penerapan pengujian *black box* umumnya menggunakan instrumen skenario uji coba yang mencakup kondisi valid maupun tidak valid. (Hasanudin, 2022) menerapkan metode ini dengan menguji kecocokan hasil pengolahan data aplikasi, seperti memasukkan parameter *username* dan *password* yang salah pada form untuk melihat apakah sistem merespons dan menolak akses sesuai dengan aturan keamanan yang ditetapkan. Pendekatan ini secara efektif memvalidasi keandalan program dan meminimalisir risiko munculnya celah fungsional saat perangkat lunak dirilis.

Pengujian *black box* adalah standar evaluasi fungsional yang andal untuk menjamin kualitas sebuah perangkat lunak sebelum diimplementasikan. Dengan mengutamakan pengujian pada mekanisme *input-output* sistem, metode ini memberikan kepastian objektif bahwa seluruh fitur beroperasi secara presisi, sesuai dengan ekspektasi, dan siap digunakan oleh pengguna.

B. Kajian Empiris

Kajian empiris terdiri dari temuan penelitian yang relevan dengan topik penelitian. Adapun penelitian terdahulu yang dijadikan acuan berkaitan dengan bahasa pemrograman Python, *framework* Streamlit, sistem basis data SQLite, serta pemodelan sistem.

Penelitian serupa yang dilakukan oleh (W. D. Putri et al., 2025) dengan judul "Pengembangan Sistem E-Commerce Berbasis Python untuk Optimasi

Penjualan dan Penyewaan Printer". Penelitian tersebut menggunakan metodologi *waterfall*. Sistem yang dikembangkan berupa platform *e-commerce* berbasis Python. Hasil penelitian menunjukkan bahwa sistem ini mampu meningkatkan jumlah transaksi, memperbaiki pengelolaan inventaris, dan meningkatkan kepuasan pelanggan secara efisien.

Penelitian yang dilakukan oleh (Syafarina, 2023) dengan judul "Implementasi Framework Streamlit Sebagai Prediksi Harga Jual Rumah Dengan Linear Regresi". Penelitian tersebut menggunakan metode kuantitatif dan algoritma regresi linear. Media yang dikembangkan berupa aplikasi web interaktif menggunakan *framework* Streamlit. Hasil penelitian menunjukkan bahwa aplikasi berbasis *Artificial Intelligence* ini mampu memberikan estimasi harga yang akurat dengan nilai akurasi sebesar 67,8% serta menyajikan antarmuka yang mudah digunakan bagi pengguna akhir.

Penelitian yang dilakukan oleh (Azhar et al., 2024) dengan judul "Prediksi Harga Mobil Audi Bekas Menggunakan Model Regresi Linear dengan Framework Streamlit". Penelitian ini menggunakan algoritma *Linear Regression*. Media yang dikembangkan memanfaatkan *framework* Streamlit untuk membangun aplikasi web menggunakan skrip Python yang sederhana. Hasil penelitian menunjukkan bahwa model memiliki nilai akurasi sebesar 94%, dan Streamlit memfasilitasi pembuatan antarmuka pengguna yang responsif dengan cepat.

Penelitian yang dilakukan oleh (Nugraha & Susetyo, 2023) dengan judul "Analisis Perbandingan Performa Database DuckDB dan SQLite pada

Pengolahan Big Data". Penelitian tersebut membandingkan performa kedua basis data dalam mengeksekusi berbagai *query*. Hasil penelitian menunjukkan bahwa SQLite terbukti sangat unggul dan lebih cepat dalam mengeksekusi *query* transaksi operasional seperti *insert*, *update* semua kolom, *delete*, dan *select* dibandingkan dengan basis data lainnya.

Penelitian yang dilakukan oleh (Azizah & Razaqi, 2026) dengan judul "Studi Tentang Penggunaan Sistem Database untuk Pengelolaan Data Pelanggan pada Usaha Kecil". Penelitian tersebut menggunakan metode deskriptif kualitatif dengan pendekatan studi kasus. Media yang dikaji adalah penerapan sistem *database* untuk menggantikan pencatatan manual. Hasil penelitian menunjukkan bahwa penerapan sistem *database* mampu meningkatkan efisiensi pengelolaan data hingga 90,3% dan menurunkan tingkat kesalahan pencatatan sebesar 83,3%.

Berdasarkan penelitian yang telah disebutkan di atas, dapat disimpulkan bahwa bahasa pemrograman Python, *framework* Streamlit, dan sistem basis data (seperti SQLite) terbukti efektif dalam meningkatkan efisiensi operasional, akurasi pengolahan data, serta penyajian antarmuka yang interaktif. Namun, penelitian yang secara khusus mengintegrasikan Python, Streamlit, dan SQLite ke dalam satu arsitektur sistem informasi yang utuh dan dimodelkan secara komprehensif menggunakan UML masih perlu diperkuat. Karena itu, penelitian ini berfokus pada pengembangan sistem informasi berbasis web yang memanfaatkan Python dan Streamlit untuk kecepatan

pembuatan antarmuka, serta SQLite sebagai basis data lokal yang ringan dan andal.

C. Kerangka Berpikir

Kerangka berpikir dalam penelitian ini berangkat dari permasalahan operasional yang terjadi di Toko Plastik T'Four, di mana manajemen inventaris dan pencatatan transaksi masih dilakukan secara manual menggunakan nota fisik. Kondisi ini mengakibatkan pemilik toko kesulitan melacak pergerakan penjualan, memicu penumpukan stok mati (*deadstock*) yang mengendap lebih dari satu tahun, serta menghambat penentuan strategi promosi yang akurat.

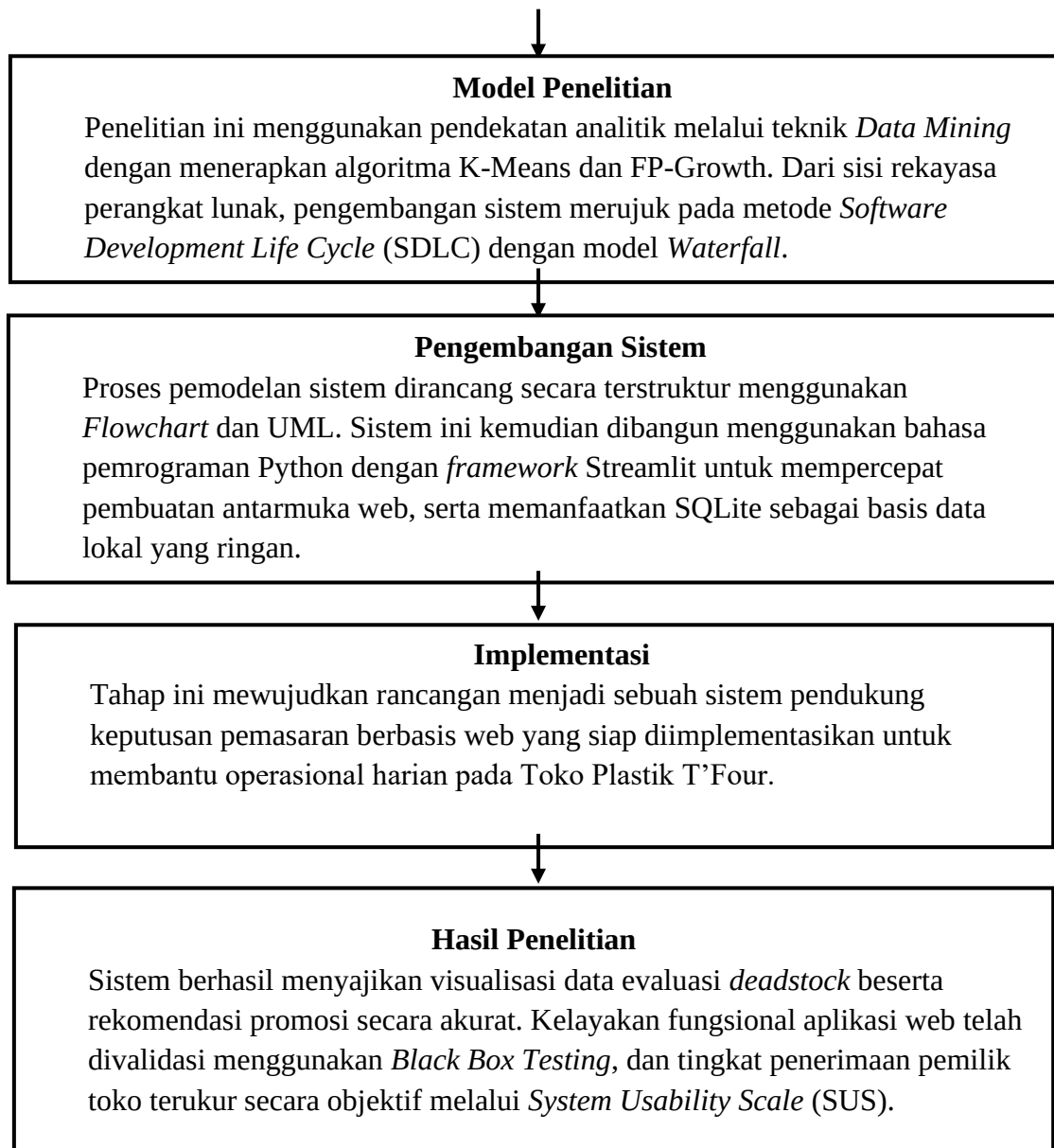
Untuk mengatasi permasalahan tersebut, penelitian ini mengusulkan penerapan teknik *data mining* yang diintegrasikan ke dalam sebuah sistem informasi berbasis web. Proses pengembangan sistem merujuk pada metode *Waterfall* dan dimodelkan secara terstruktur menggunakan *Flowchart* serta *Unified Modeling Language* (UML). Sistem ini dibangun menggunakan bahasa pemrograman Python dengan *framework* Streamlit untuk menghasilkan antarmuka yang cepat dan interaktif, serta menggunakan SQLite sebagai basis data lokal. Dari sisi analitik, algoritma K-Means digunakan untuk mengelompokkan dan mendeteksi produk *deadstock*, sementara algoritma FP-Growth diimplementasikan untuk menemukan pola asosiasi pembelian sebagai dasar rekomendasi paket *bundling* atau diskon.

Sebagai tahap akhir, aplikasi web yang telah dibangun akan diuji kelayakannya menggunakan *Black Box Testing* untuk memvalidasi fungsionalitas sistem dan *System Usability Scale* (SUS) untuk mengukur tingkat

penerimaan dan kepuasan pemilik toko. Hasil akhir dari penelitian ini diharapkan dapat memberikan solusi digital berupa laporan dan visualisasi data yang memudahkan pemilik Toko Plastik T'Four dalam meminimalkan kerugian akibat stok stagnan dan mengoptimalkan strategi pemasaran.

Rumusan Masalah

1. Pencatatan arus transaksi di toko masih dilakukan secara manual menggunakan tumpukan nota fisik kertas.
2. Terjadi penumpukan barang stagnan (*deadstock*) lebih dari satu tahun yang sering kali tidak disadari oleh pemilik.
3. Penentuan strategi promosi dan pembuatan paket *bundling* produk belum didasarkan pada analisis data transaksi.



Gambar 2. 1 Kerangka Berpikir