BAB II
LANDASAN TEORI

Bab ini memaparkan berbagai teori yang menjadi dasar penelitian, meliputi konsep sistem informasi, pengertian aplikasi POS, metode Waterfall, bahasa pemrograman Visual Basic .NET, database Microsoft Access, serta diagram UML seperti *Use Case*, *Activity Diagram*, dan *Class Diagram*.

2.1 Kajian Teori

2.1.1 Sistem Informasi

Sistem dapat diartikan sebagai suatu kesatuan yang tersusun atas berbagai elemen yang saling terkait dan berkolaborasi secara harmonis untuk mewujudkan suatu sasaran tertentu dalam lingkungan yang dinamis (Oetari S, 2025) Secara lebih spesifik, Sistem adalah serangkaian prosedur yang saling berhubungan dan tersusun secara terpadu guna menjalankan suatu kegiatan atau mencapai sasaran yang telah ditetapkan (Pratama, 2021). Di samping itu, sistem juga dapat dimaknai sebagai perpaduan berbagai elemen yang saling bergantung dan berkolaborasi dalam memproses masukan yang diterima, mengolahnya melalui serangkaian proses, hingga menghasilkan keluaran yang diinginkan (Efendi, 2021)

Informasi merupakan produk dari proses pengolahan data yang mampu memberikan makna, pengetahuan, atau gambaran yang jelas mengenai suatu topik atau kondisi tertentu (Oetari S, 2025). Informasi juga dapat dimaknai sebagai data yang telah melalui proses pengolahan sehingga menjadi lebih berarti dan memiliki kegunaan bagi penerimanya, karena mampu menggambarkan kejadian nyata serta menjadi dasar dalam pengambilan keputusan (Pratama, 2021). Sistem informasi merupakan suatu mekanisme di dalam organisasi yang memadukan berbagai kebutuhan pemrosesan transaksi sehari-hari dengan fungsi operasional dan manajerial, sehingga mampu menyajikan laporan yang dibutuhkan oleh berbagai pihak yang berkepentingan (Oetari S, 2025). Sistem informasi merupakan gabungan dari berbagai elemen yang bekerja sama secara sinergis, baik secara

manual maupun terkomputerisasi, untuk melaksanakan pengolahan data seperti pengumpulan, penyimpanan, hingga pemrosesan, guna menghasilkan informasi yang berarti dan berguna dalam pengambilan keputusan (Efendi, 2021).

2.1.2 Aplikasi

Aplikasi adalah perangkat lunak yang dibuat oleh suatu perusahaan teknologi dengan tujuan membantu pengguna dalam menyelesaikan berbagai pekerjaannya (Oetari S, 2025). Secara lebih rinci, Aplikasi merupakan perangkat lunak yang di dalamnya terdapat berbagai fungsi maupun instruksi yang dapat digunakan oleh pengguna untuk menyelesaikan pekerjaan-pekerjaan tertentu (Martono & Zulfi, 2022). Aplikasi juga dapat diartikan sebagai program yang dikembangkan untuk maksud tertentu, di mana pengguna dapat menjalankan berbagai aktivitas atau instruksi sesuai dengan kemampuan yang telah tersedia di dalamnya, sehingga program tersebut dapat bekerja sebagaimana yang diharapkan penggunanya (Martono & Zulfi, 2022). Berdasarkan platform penggunaannya, aplikasi dikelompokkan menjadi tiga kategori utama, yaitu aplikasi desktop yang dioperasikan pada perangkat komputer, aplikasi berbasis web yang diakses melalui peramban, serta aplikasi bergerak (*mobile*) yang dioperasikan pada perangkat portabel seperti ponsel cerdas atau tablet (Oetari S, 2025).

2.1.3 *Point Of Sale* (POS)

Point Of Sale (POS) adalah sebuah sistem informasi yang dibangun untuk mendukung kelancaran proses transaksi, yang di dalamnya mencakup pemanfaatan mesin kasir (Apriyani et al., 2022). Secara umum, POS dapat diartikan sebagai suatu sistem yang memungkinkan berlangsungnya proses transaksi penjualan (Efendi, 2021). Sistem ini tidak sekadar berperan dalam pencatatan transaksi penjualan dan pembelian, melainkan juga terhubung dengan sejumlah modul lainnya, seperti perhitungan akuntansi, pengelolaan inventaris, manajemen stok, penggajian tenaga kerja, serta pencatatan piutang. (Apriyani et al., 2022).

Aplikasi POS berperan dalam mendukung kelancaran transaksi barang dan jasa, sekaligus meningkatkan keamanan serta keakuratan data, termasuk dalam pencatatan persediaan barang (Oetari S, 2025). Kegiatan POS berorientasi pada

penjualan dan berfungsi sebagai sistem pendukung dalam proses transaksi (Arman & Maberur, 2022). Setiap sistem POS terdiri dari dua komponen utama, yaitu perangkat keras yang mencakup terminal atau PC, printer nota, laci kasir, serta perangkat pembayaran, dan perangkat lunak seperti sistem manajemen inventaris, pelaporan, pembelian, pengelolaan pelanggan, serta fitur keamanan transaksi Kedua unsur ini saling bersinergi dan berpadu dalam setiap tahapan transaksi. (Oetari S, 2025). Sistem POS memiliki peran strategis dalam dunia bisnis karena menjadi titik utama penerimaan pembayaran dari pelanggan, yang merupakan indikator paling sederhana bagi pemilik usaha untuk mengukur pendapatan (Arman & Maberur, 2022).

Sistem POS yang baik mampu menyajikan informasi transaksi dan laporan penjualan secara real-time (Arman & Maberur, 2022). Selain itu, POS dapat memberikan kemudahan bagi pemilik toko dalam mengatur aktivitas penjualan serta mengawasi ketersediaan stok barang (Irwan et al., 2023). Dengan kemampuan merekam transaksi dalam bentuk digital, mengatur stok barang secara otomatis, serta menghadirkan laporan penjualan secara terstruktur yang up-to-date, POS berkontribusi pada peningkatan efisiensi operasional dan kualitas pelayanan (Herdiyanto & Yunita, 2024).

2.1.4 Visual Basic .NET

Visual Basic .NET (VB.Net) adalah salah satu bahasa pemrograman yang menerapkan konsep berbasis Object Oriented Programming (OOP) yang diciptakan oleh Microsoft dan menjadi bagian dari lingkungan .NET Framework (Fiby et al., 2024). Bahasa ini dirancang dengan pendekatan berbasis objek dan dioperasikan di atas .NET framework, sehingga memerlukan dukungan perangkat lunak .NET untuk dapat dioperasikan. Dengan demikian, VB.Net memiliki akses penuh terhadap berbagai pustaka (libraries) yang tersedia dalam .NET framework (Ginting, 2022). Keunggulan utama VB.Net antara lain adalah waktu waktu yang dibutuhkan untuk mempelajari dan mengembangkannya tergolong singkat, sehingga sangat sesuai untuk digunakan dalam pembuatan beragam jenis aplikasi. Bahasa ini juga dilengkapi dengan fitur wizard yang memudahkan proses

pengembangan aplikasi. Selain itu, VB.Net memiliki kemampuan untuk berintegrasi dengan layanan transaksi server dari Microsoft, mendukung pembuatan server otomasi ActiveX, serta dapat terhubung dengan jaringan internet (Ginting, 2022).

Dalam penelitian ini, Visual Basic .NET digunakan untuk merancang antarmuka pengguna (*user interface*) dan mengembangkan logika pemrograman yang mendukung transaksi penjualan di Toko Martini. Penggunaan VB.Net dalam aplikasi POS telah terbukti berhasil pada penelitian (Ginting, 2022) di Apotek Thamrin dan penelitian (Aritonang et al., 2023) di Toko Mutiara.

2.1.5 Database Microsoft Access

Microsoft Access merupakan sebuah sebuah sistem pengelolaan basis data relasional (*Relational Database Management System* / RDBMS) yang diciptakan oleh Microsoft. Perangkat lunak ini termasuk dalam paket Microsoft Office dan dirancang khusus untuk memenuhi kebutuhan pengelolaan data pada skala kecil hingga menengah(Ginting, 2022). Sementara itu, MySQL merupakan salah satu implementasi dari konsep basis data *Structured Query Language* (SQL). SQL sendiri merupakan bahasa baku yang dimanfaatkan untuk mengoperasikan basis data, terutama dalam proses seleksi dan pemasukan data, yang dirancang untuk memudahkan manipulasi data secara otomatis dan efisien (Efendi, 2021).

Penelitian Ginting (2022) di Apotek Thamrin menggunakan Microsoft Access sebagai database untuk menyimpan data informasi obat, data pemasok, transaksi pembelian, serta data penjualan. Penelitian Aritonang, Dauli, dan Martyani (2023) juga menggunakan Microsoft Access sebagai database dalam pengembangan sistem informasi penjualan di Toko Mutiara. Dalam penelitian ini, Microsoft Access digunakan karena sifatnya yang ringan, tidak memerlukan server terpisah, mudah diinstal dan di-backup, serta cocok untuk toko dengan spesifikasi komputer rendah (Ginting, 2022).

Penelitian yang dilakukan oleh Mair dan Sari (2021) di Adibah Boutique juga menggunakan database Access dalam pengembangan aplikasi kasir berbasis desktop. Penelitian tersebut menunjukkan bahwa database Access mampu

mendukung proses transaksi penjualan, manajemen inventaris, dan penyusunan laporan dengan baik(Mair & Sari, 2021).

2.1.6 *Crystal Report*

Crystal Report merupakan perangkat lunak yang digunakan untuk menyusun laporan (*reporting tool*) dan terintegrasi secara langsung dengan Visual Studio .NET. Penyusunan laporan menggunakan *Crystal Report* tergolong mudah karena tidak membutuhkan banyak baris kode dan prosesnya relatif sederhana. Selain itu, perangkat ini juga dilengkapi dengan fitur ekspor yang mendukung beragam format file seperti PDF, HTML, dan Microsoft Office (Ginting, 2021). Keunggulan *Crystal Report* antara lain integrasi dengan Visual Studio, desain laporan yang fleksibel, kemampuan ekspor ke berbagai format, serta kemampuan mengambil data dari berbagai sumber termasuk Microsoft Access dan MySQL (Ginting, 2021)

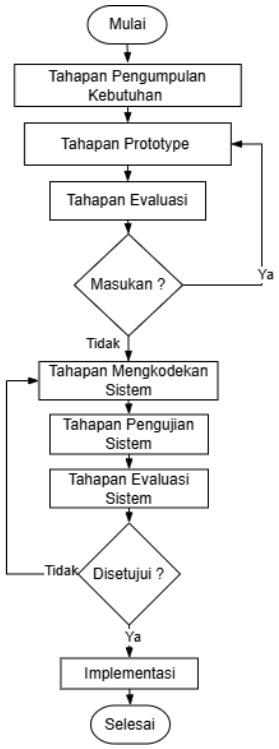
Penelitian Ginting (2021) di Varen Coffee menggunakan *Crystal Report* untuk membuat laporan penjualan, di mana hasilnya laporan penjualan dapat ditampilkan dan dicetak dengan rapi serta profesional. Dalam penelitian ini, *Crystal Report* akan dimanfaatkan untuk mencetak laporan penjualan harian maupun laporan penjualan bulanan, dan laporan data barang di Toko Martini.

2.1.7 Metode *Prototype*

Metode *Prototype* adalah pendekatan pengembangan perangkat lunak di mana pembuat sistem dan pengguna saling berinteraksi untuk membuat contoh (*Prototype*) awal dari sistem yang diinginkan (Herdiyanto & Yunita, 2024). Pendekatan prototyping diawali dengan proses mendengarkan secara aktif terhadap keperluan serta masukan yang diberikan oleh pengguna. Dalam tahap ini, pihak pengembang dan pengguna berdiskusi secara langsung untuk menetapkan tujuan umum dari perangkat lunak yang akan dikembangkan serta mengidentifikasi seluruh persyaratan yang diperlukan (Herdiyanto & Yunita, 2024). Selanjutnya, pengembang menyusun suatu gambaran awal mengenai aplikasi yang akan dibangun. Gambaran ini kemudian dipresentasikan kepada pengguna atau pelanggan, dengan fokus utama pada representasi visual dari bagian-bagian aplikasi

yang akan tampak dan dirasakan langsung oleh pengguna nantinya (Herdiyanto & Yunita, 2024).

Tahapan metode *Prototype* meliputi: tahapan pengumpulan kebutuhan, tahapan pembuatan *Prototype* , tahapan evaluasi *Prototype* , tahapan mengkodekan sistem, tahapan pengujian system, tahapan evaluasi sistem implementasi sistem(Herdiyanto & Yunita, 2024). Metode *Prototype* menawarkan keunggulan utama berupa proses perancangan yang lebih adaptif dan tanggap terhadap perubahan keinginan pengguna. Pengembang dapat menyusun versi permulaan aplikasi yang selanjutnya diujicobakan serta dinilai secara langsung oleh pengguna. Dengan demikian, masukan yang diberikan dapat segera diakomodasi untuk menyempurnakan fitur dan fungsionalitas sistem secara berkelanjutan (Herdiyanto & Yunita, 2024). Hasil akhir yang dicapai dari penerapan metode ini menghasilkan aplikasi yang lebih sesuai dengan kebutuhan operasional, mampu meningkatkan efisiensi dalam proses transaksi dan pencatatan data, serta memberikan pengalaman penggunaan yang lebih baik bagi pengguna (Herdiyanto & Yunita, 2024).



Gambar 2. 1 Tahapan *Prototype*

Penelitian Herdiyanto, dan Yunita (2025) menggunakan metode *Prototype* dalam merancang aplikasi pada pengembangan sistem sistem POS berbasis web yang diterapkan di Kantin STMIK Widya Cipta Dharma menunjukkan bahwa penerapan metode *Prototype* memberikan fleksibilitas yang tinggi dalam proses perancangan. Metode ini terbukti mampu menyesuaikan diri secara dinamis terhadap kebutuhan pengguna, sehingga hasil akhir sistem lebih selaras dengan keinginan dan tuntutan operasional pengguna. Dalam penelitian ini, metode *Prototype* dipilih karena memungkinkan pemilik Toko Martini untuk melihat langsung contoh awal aplikasi sebelum pengembangan penuh dilakukan, sehingga masukan dapat diberikan sejak dini.

Selain penelitian Charlos Herdiyanto dkk (2025), metode *Prototype* juga diakui sebagai pendekatan yang fleksibel dalam pengembangan sistem informasi. Metode ini memungkinkan pengembang untuk berkolaborasi secara intensif dengan pengguna sehingga menghasilkan sistem yang betul-betul cocok dengan kebutuhan (Prayogo et al., 2025).

1. Perbandingan Metode Waterfall dan *Prototype*

Selain metode *Prototype* , pendekatan lain yang banyak dimanfaatkan dalam pengembangan perangkat lunak adalah metode Waterfall. Model ini merupakan pendekatan pengembangan sistem informasi yang bersifat terstruktur dan bertahap (Efendi, 2021). Setiap tahapan dalam metode ini wajib diselesaikan seluruhnya terlebih dahulu sebelum melangkah ke tahap selanjutnya, sehingga proses pengembangannya berlangsung secara linear dan terstruktur (Pratama, 2021).

Beberapa penelitian yang menggunakan metode Waterfall dalam pengembangan sistem *Point Of Sale* antara lain penelitian Munandar dkk. (2023) yang merancang aplikasi kasir berbasis desktop yang ditujukan untuk toko kelontong, penelitian Aritonang, Dauli, & Martyani (2023) tentang pengembangan sistem informasi penjualan pada Toko Mutiara, serta penelitian Maridaningsih, Setiawan, & Nugroho (2025) tentang perancangan sistem POS yang bertujuan untuk meningkatkan efektivitas dalam mengelola penjualan dan persediaan barang. Berikut adalah perbandingan antara metode Waterfall dan *Prototype* :

Tabel 2. 1 Pembanding Metode

Aspek	**Metode Waterfall**	**Metode *Prototype***
Pendekatan	Linear dan berurutan (Pratama, 2021)	Iteratif dan siklus umpan balik (Herdiyanto, Nursobah, & Yunita, 2025)
Keterlibatan Pengguna	Terbatas pada tahap awal (Munandar dkk., 2023)	Aktif sepanjang proses pengembangan (Prayudha, Irwansyah, & Anra, 2024)
Fleksibilitas	Rendah (sulit mengubah kebutuhan) (Ginting, 2021)	Tinggi (perubahan dapat diakomodasi) (Herdiyanto, Nursobah, & Yunita, 2025)
Dokumentasi	Lengkap dan terstruktur (Oetari, 2025)	Minimal, fokus pada pengembangan cepat (Prayudha, Irwansyah, & Anra, 2024)
Waktu Pengembangan	Relatif lama (Ginting, 2022)	Relatif singkat (Herdiyanto, Nursobah, & Yunita, 2025)
Biaya	Lebih mahal (Apriyani dkk., 2022)	Lebih hemat (Oetari, 2025)
Kesesuaian	Sistem generik dengan kebutuhan jelas (Arman & Maberur, 2022)	Sistem *custom* dengan kebutuhan dinamis (Prayudha, Irwansyah, & Anra, 2024)
Resiko Kesalahan	Kesalahan pemahaman kebutuhan tinggi (Maridaningsih, Setiawan, & Nugroho, 2025)	Risiko lebih rendah karena umpan balik terus-menerus (Herdiyanto, Nursobah, & Yunita, 2025)

2. Alasan Pemilihan Metode *Prototype* dalam Penelitian Ini

Berdasarkan perbandingan di atas, penelitian ini memilih menggunakan metode *Prototype* dengan pertimbangan sebagai berikut:

1. Kebutuhan pengguna yang belum sepenuhnya jelas di awal. Pemilik Toko Martini belum memiliki gambaran detail tentang aplikasi POS yang diinginkan. Metode *Prototype* memungkinkan pengguna melihat contoh awal aplikasi dan memberikan masukan secara bertahap (Herdiyanto & Yunita, 2024).
2. Fleksibilitas terhadap perubahan. Selama proses pengembangan, pemilik toko dapat memberikan umpan balik dan perubahan kebutuhan dapat diakomodasi dengan mudah, berbeda dengan Waterfall yang kaku (Prayogo et al., 2025).
3. Pengembangan lebih cepat dan hemat biaya. Metode *Prototype* memungkinkan pengembangan yang lebih cepat dengan biaya yang lebih rendah, sesuai dengan skala toko kelontong kecil (Oetari S, 2025).
4. Sistem yang bersifat *customize*. Aplikasi POS ini dibuat berdasarkan permintaan dan kebutuhan spesifik Toko Martini, bukan sistem generik yang bisa diterapkan di mana saja (Herdiyanto & Yunita, 2024).
5. Komunikasi yang lebih baik. *Prototype* memungkinkan pengembang dan pengguna berkomunikasi secara efektif melalui visualisasi awal sistem, sehingga risiko kesalahan pemahaman kebutuhan dapat diminimalisir (Prayogo et al., 2025).

Dengan demikian, metode *Prototype* dinilai lebih sesuai untuk pengembangan aplikasi POS di Toko Martini karena memberikan fleksibilitas, efisiensi, dan keterlibatan pengguna yang lebih baik dibandingkan metode Waterfall .(Oetari S, 2025).

2.1.8 *Unified Modelling Language* (UML)

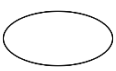
Unified Modelling Language (UML) adalah bahasa standar yang dimanfaatkan untuk mendokumentasikan, menspesifikasikan, serta membangun

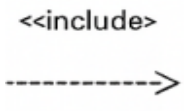
perangkat lunak (Oetari S, 2025). UML juga berperan sebagai metodologi dalam pengembangan sistem berorientasi objek serta menjadi sarana pendukung yang mendukung proses pengembangan sistem secara keseluruhan (Oetari S, 2025). Penerapan UML biasanya ditujukan untuk berbagai keperluan, seperti perancangan perangkat lunak, media komunikasi antara sistem dan proses bisnis, serta alat untuk menganalisis dan menjabarkan kebutuhan sistem secara rinci. UML juga digunakan untuk mendokumentasikan sistem yang sedang berjalan beserta prosedur dan struktur organisasinya (Pratama, 2021). Selain itu, UML banyak igunakan di lingkungan industri untuk merumuskan kebutuhan (*requirement*), menyusun analisis dan perancangan, serta memvisualisasikan arsitektur sistem dalam pengembangan berbasis objek (Salsabilla Oetari, 2025).

1. *Use Case Diagram*

Use Case Diagram adalah salah satu tipe diagram yang menunjukkan seluruh kemampuan yang disediakan oleh suatu sistem serta aktor-aktor yang terlibat di dalamnya (Oetari S, 2025). Diagram ini memusatkan perhatian pada apa yang dikerjakan oleh sistem, bukan pada cara sistem tersebut menjalankan fungsinya (Oetari S, 2025). Di samping itu, *Use Case Diagram* juga memvisualisasikan hubungan interaksi antara aktor (pengguna) dan sistem secara visual, sehingga dapat menyajikan gambaran yang tegas tentang cakupan dan batas-batas sistem yang sedang dibangun (Ginting, 2022).

Tabel 2. 2 *Use Cas Diagram*

No	Simbol	Deskripsi
1.		Aktor adalah pihak yang berinteraksi langsung dengan sistem, baik sebagai pengguna maupun entitas eksternal yang memberi input atau menerima output sistem.
	Use Cas 	*Use Case* adalah representasi dari fungsi atau layanan yang disediakan oleh sistem untuk dimanfaatkan oleh aktor.

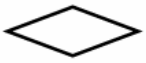
No	Simbol	Deskripsi
	Association	Association adalah relasi aktor-*Use Case* yang menunjukkan interaksi langsung pengguna-sistem dan akses aktor terhadap fungsi tertentu.
	Extend 	Extend adalah jenis hubungan dalam *Use Case Diagram* yang menggambarkan bahwa suatu *Use Case* dapat diperluas dengan perilaku tambahan dari *Use Case* lain, namun penambahan tersebut tidak wajib dilakukan (opsional).
	Generalization 	Generalization dalam *Use Case Diagram* adalah hubungan yang menunjukkan adanya pewarisan (inheritance) antara aktor atau antar *Use Case*.
		Include dalam *Use Case Diagram* adalah hubungan yang menunjukkan bahwa suatu *Use Case* selalu menyertakan atau memanggil *Use Case* lain sebagai bagian dari prosesnya.
	System Boundary 	System Boundary menunjukkan batasan ruang lingkup dari sistem yang sedang dibangun. Semua *Use Case* ditempatkan di dalam batas tersebut, sedangkan aktor berada di luar sistem dan berinteraksi dari luar

Penerapan *Use Case Diagram* dalam perancangan sistem POS telah banyak dilakukan oleh berbagai peneliti. Beberapa di antaranya adalah penelitian oleh Ginting (2021) pada sistem penjualan di Varen Coffee, Pratama (2021) di Toko Aula Collection, Efendi (2021) di Warung Batak Iwan, serta Oetari (2025) di Toko DSAWD Collection. Sejalan dengan studi-studi sebelumnya, pada penelitian ini *Use Case Diagram* akan dimanfaatkan untuk memodelkan hubungan timbal balik antara dua pelaku utama, yaitu admin dan kasir, dengan sistem POS yang dikembangkan untuk Toko Martini.

2. *Activity Diagram*

Activity Diagram merupakan salah satu jenis pemodelan yang digunakan dalam sistem untuk menunjukkan alur aktivitas yang terjadi di dalamnya tersebut (Oetari S, 2025). Diagram ini dipakai untuk menggambarkan proses-proses yang berlangsung dalam program tanpa harus melihat kode sumber maupun tampilan antarmuka(Oetari S, 2025). *Activity Diagram* menggambarkan berbagai alur kegiatan yang dirancang dalam sistem, termasuk bagaimana masing-masing alur diawali, pilihan-pilihan yang dapat muncul selama proses berjalan, serta bagaimana alur tersebut berakhir (Efendi, 2021). Diagram ini juga menunjukkan berbagai alur yang dimulai dan keputusan yang akan diambil, serta menggambarkan proses paralel yang dapat terjadi selama beberapa eksekusi yang telah dirancang (Oetari S, 2025).

Tabel 2. 3 *Activity Diagram*

No	Simbol	Deskripsi
1.	Status Awal 	**Status awal (Initial Node)** merupakan titik dimulainya suatu proses dalam *Activity Diagram*. Simbol ini menandakan awal dari alur aktivitas yang akan dijalankan oleh sistem atau pengguna.
2.	Aktivitas 	**Aktivitas (Activity)** adalah proses atau kegiatan yang dilakukan oleh sistem. Biasanya dituliskan dalam bentuk kata kerja seperti login, input data, atau simpan data.
3.	Percabangan 	**Decision** adalah simbol yang digunakan untuk menunjukkan percabangan proses berdasarkan kondisi tertentu, sehingga menghasilkan lebih dari satu alur.
4.	Penggabungan/Join	**Join** adalah simbol yang digunakan untuk menggabungkan beberapa alur proses yang sebelumnya bercabang menjadi satu alur Kembali.

No	Simbol	Deskripsi
5.	Status Akhir 	**Status akhir (Final Node)** adalah simbol yang menunjukkan berakhirnya suatu proses dalam sistem. Diagram aktivitas umumnya memiliki satu titik akhir sebagai penutup alur.

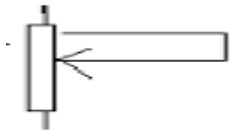
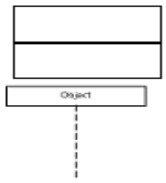
Activity Diagram telah digunakan dalam perancangan aplikasi POS oleh Ginting (2021), Pratama (2021), Efendi (2021), Penelitian yang dilakukan oleh Rahma et al (2025), Oetari (2025). Dalam penelitian ini, *Activity Diagram* akan digunakan untuk menggambarkan alur transaksi penjualan, alur login, dan alur pengelolaan data di Toko Martini.

3. *Sequence Diagram*

Sequence Diagram adalah diagram UML untuk memvisualisasikan urutan interaksi antar objek berdasarkan waktu. Diagram ini menggambarkan alur pengiriman dan penerimaan pesan antar objek, sehingga memudahkan dalam memahami pola komunikasi yang terjadi di dalam sistem.

Tabel 2. 4 Sequence Diagram

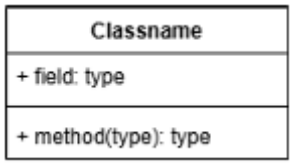
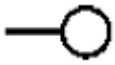
No	Simbol	Deskripsi
1.		Actor adalah pihak atau pengguna yang berinteraksi langsung dengan sistem. Actor dapat berupa manusia, perangkat lain, atau sistem lain yang memberikan input maupun menerima output dari sistem.
2.	Object Masage 	Object Message merupakan pesan atau komunikasi yang terjadi antar objek dalam sistem. Simbol ini digunakan untuk menunjukkan proses pertukaran informasi serta urutan aktivitas yang berlangsung dari satu objek ke objek lainnya.
3.	Masage to Self	Message to Self adalah pesan yang dikirim oleh suatu objek kepada dirinya sendiri. Simbol ini menunjukkan bahwa objek sedang menjalankan

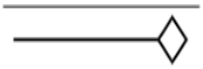
No	Simbol	Deskripsi
		proses internal atau memanggil fungsi yang masih berada dalam objek yang sama.
4.	Lifeline 	Lifeline menggambarkan keberadaan suatu objek selama proses interaksi berlangsung. Simbol ini berbentuk garis vertikal putus-putus yang menunjukkan waktu hidup objek ketika berinteraksi dengan objek lain dalam *Sequence Diagram*.

4. *Class Diagram*

Class Diagram adalah diagram UML untuk memvisualisasikan struktur sistem melalui kelas, atribut, metode, dan relasi antarkelas. Diagram ini memegang peranan penting dalam fase perancangan karena menjadi landasan bagi pembuatan basis data dan pengembangan program.

Tabel 2. 5 *Class Diagram*

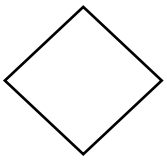
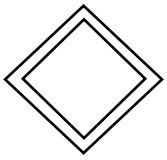
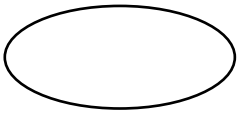
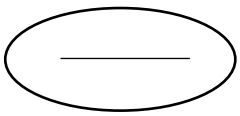
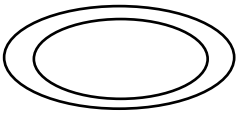
No	Simbol	Deskripsi
1.		Simbol ini digunakan untuk menggambarkan **struktur kelas dalam sistem.**
2.	Antarmuka/Interface 	Memiliki konsep yang sama dengan interface pada pemrograman berorientasi objek.
3.	Asosiasi/Association	Relasi antarkelas dengan makna umum, asosiasi biasanya juga disertai dengan multiplicity.
4.	Asosiasi berarah / Directed Association	Relasi antarkelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi biasanya juga disertai dengan multiplicity.

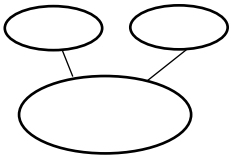
No	Simbol	Deskripsi
5.	Generalisasi 	Relasi antarkelas generalisasi dengan makna spesialisasi khusus).
6.	Kebergantungan / Dependency 	Relasi antarkelas dengan makna kebergantungan antarkelas.
7.	Composition 	Relasi antarkelas (whole-part).

2.1.9 *Entity Relationship Diagram* (ERD)

Perancangan basis data sering kali memanfaatkan *Entity Relationship Diagram* (ERD) sebagai metode pemodelan utamanya dengan cara mengidentifikasi elemen-elemen data penting serta menggambarkannya ke dalam suatu model konseptual (Ginting, 2022). ERD Diagram ini berfungsi sebagai alat pemodelan yang membantu dalam memahami secara mendalam struktur data beserta pemanfaatannya dalam lingkungan organisasi atau perusahaan (Ginting, 2022). ERD memiliki peran penting dalam menggambarkan hubungan antar entitas serta mengidentifikasi karakteristik dari setiap entitas tersebut. Dengan demikian, representasi ini dapat dengan mudah dikonversi menjadi tabel-tabel database yang nantinya akan dipakai sebagai media penyimpanan data secara terstruktur (Prayudha et al., 2024). Secara umum, ERD terdiri dari tiga komponen dasar, yaitu entitas yang merupakan objek utama, atribut yang berisi informasi terkait entitas, serta relasi yang menggambarkan keterhubungan antar entitas. Relasi ini dapat memiliki berbagai jenis, seperti relasi satu ke satu, satu ke banyak, maupun banyak ke banyak (Prayudha et al., 2024).

Tabel 2. 6 Diagram ERD

No	Simbol	Nama Simbol	Fungsi / Keterangan
1		Entity	Mewakili objek atau konsep dunia nyata yang memiliki keberadaan secara independen dan dapat dibedakan dari objek lainnya. Dalam database, entity biasanya menjadi sebuah tabel. Contoh: Barang, Supplier, Karyawan.
2		Weak Entity	Entitas yang tidak memiliki atribut yang cukup untuk membentuk primary key sendiri. Keberadaannya bergantung pada entitas lain (strong entity) dan diidentifikasi melalui foreign key. Contoh: Detail Penjualan (bergantung pada Penjualan).
3		Relationship	Menunjukkan hubungan atau relasi antara dua entitas atau lebih. Relasi ini menjelaskan bagaimana entitas saling terhubung. Contoh: Supplier "memasok" Barang.
4		Identifying Relationship	Relasi yang menghubungkan weak entity dengan strong entity (entitas induk). Relasi ini menjadi bagian dari primary key weak entity. Contoh: Detail Penjualan "terkait dengan" Penjualan.
5		Atribut	Properti atau karakteristik yang dimiliki oleh suatu entitas. Atribut menggambarkan informasi detail dari entitas. Contoh: Nama Barang, Harga Jual, Alamat Supplier.
6		Atribut Primary Key	Atribut yang berfungsi sebagai pengidentifikasi unik untuk setiap record dalam suatu entitas. Primary key tidak boleh memiliki nilai yang sama (unik) dan tidak boleh kosong. Contoh: ID_BARANG, ID_SUPPLIER.
7		Atribut Multivalue	Atribut yang dapat memiliki lebih dari satu nilai untuk satu entitas. Contoh: Nomor Telepon (seorang karyawan bisa memiliki lebih dari satu nomor telepon).

No	Simbol	Nama Simbol	Fungsi / Keterangan
8		Atribut Composite	Atribut yang terdiri dari beberapa atribut yang lebih kecil (sub-atribut). Contoh: Alamat yang terdiri dari Jalan, Kota, Kode Pos.
9		Atribut Derivatif	Atribut yang nilainya diperoleh atau dihitung dari atribut lain. Nilai atribut ini tidak disimpan langsung, tetapi dihitung saat dibutuhkan. Contoh: Total (dihitung dari Jumlah × Harga Satuan), Hasil (dihitung dari Subtotal - Modal).

Penelitian Penelitian yang dilakukan oleh Apriyani et al (2022) di Qini Mart Tasikmalaya menggunakan ERD untuk merancang basis data aplikasi POS. Penelitian Zuldesfianti (2022) di Rumah Makan Baru Jaya juga menggunakan ERD dalam perancangan sistem POS. Demikian pula penelitian Oetari (2025) di Toko DSAWD Collection menggunakan ERD untuk merancang struktur seluruh basis data. Dalam penelitian ini, ERD untuk merancang struktur database Access yang terdiri dari tabel barang, transaksi, detail transaksi, dan pengguna.

2.1.10 *Black Box Testing*

Black Box Testing adalah salah satu metode pengujian perangkat lunak yang lebih mengutamakan pengecekan terhadap fungsionalitas sistem tanpa perlu memahami struktur logika maupun kode di baliknya(Fiby et al., 2024). Pendekatan ini dilakukan untuk memastikan bahwa setiap masukan dan keluaran yang dihasilkan oleh sistem telah sesuai dengan kriteria yang telah ditentukan. Pada pelaksanaannya, pengujian ini dijalankan dengan membuat sejumlah skenario uji yang dirancang untuk menguji seluruh fitur yang tersedia, guna menilai kesesuaiannya dengan kebutuhan yang diharapkan (Pratama, 2021). Pengujian *black box* dilakukan dengan membuat kasus uji yang bersifat mencoba semua fungsi dengan memakai perangkat lunak apakah sesuai dengan spesifikasi yang dibutuhkan (Pratama, 2021).

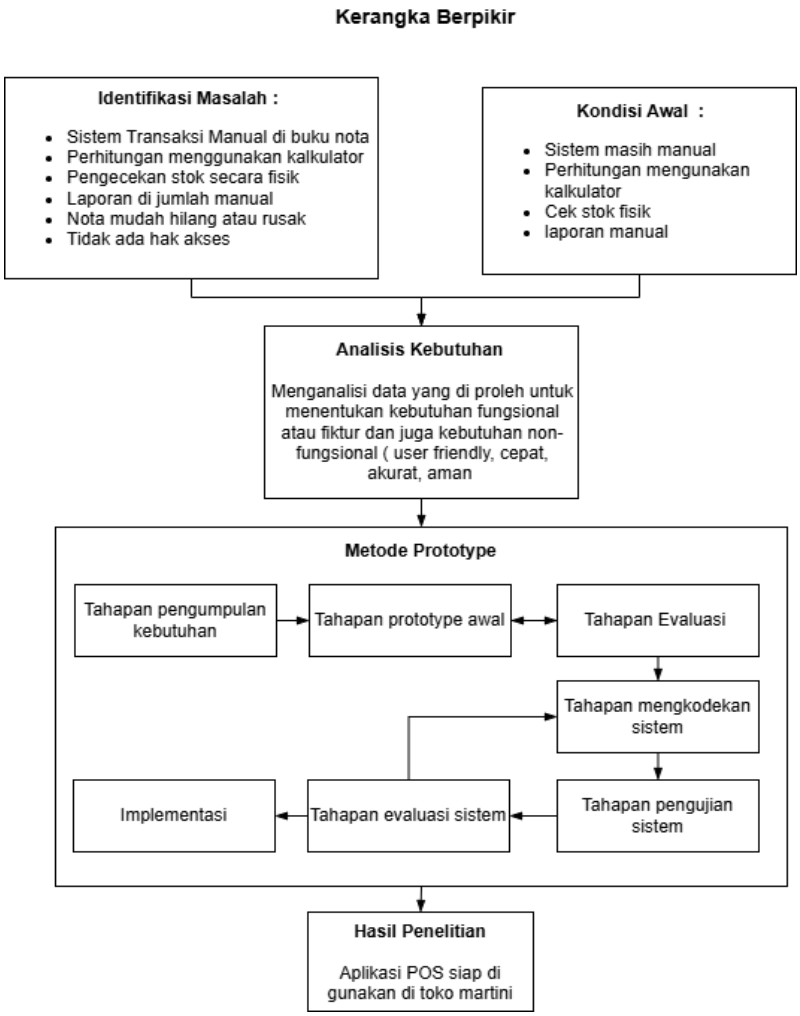
Berbagai penelitian di bidang pengembangan sistem POS telah memanfaatkan metode ini. Misalnya, Rahma dkk (2025) menerapkan pengujian

black box pada sistem POS berbasis Java Desktop di Toko Berkah Abadi dan memperoleh hasil bahwa seluruh fungsi berjalan sebagaimana mestinya (Inkud Daifatur Rahma et al., 2025). Penelitian lain oleh Pujiaistuti dkk (2023) juga menggunakan pendekatan serupa untuk menguji fungsionalitas sistem POS di Koperasi Avicenna, meskipun metode pengembangan yang digunakan berbeda, yaitu Dynamic Probability (Cholifah et al., 2023). Sementara itu, Zuldesfianti (2022) turut menerapkan metode ini pada pengembangan sistem POS berbasis desktop di Rumah Makan Baru Jaya dan menyimpulkan bahwa fitur-fitur seperti pemesanan, pembayaran, dan pelaporan berfungsi dengan baik (Zuldesfianti, 2023).

Keunggulan dari pendekatan ini adalah penguji tidak memerlukan pemahaman mendalam mengenai bahasa pemrograman, sebab pengujian dilaksanakan berdasarkan perspektif pengguna akhir. Hal ini memungkinkan terungkapnya potensi kesalahan dalam spesifikasi sistem. Selain itu, metode ini menempatkan pengembang dan penguji dalam posisi saling melengkapi dalam proses verifikasi sistem (Efendi, 2021). Dalam penelitian ini, *Black Box Testing* akan digunakan untuk menguji seluruh komponen aplikasi POS di Toko Martini, termasuk mekanisme pembuatan laporan menggunakan *Crystal Report*.

2.2 Kerangka Berpikir

Kerangka berpikir dalam penelitian ini disusun sebagai landasan logis yang menghubungkan permasalahan yang diidentifikasi, solusi yang ditawarkan, serta tahapan penelitian yang akan dilakukan untuk mencapai tujuan yang diharapkan. Kerangka ini berfungsi sebagai alur pemikiran yang sistematis untuk memudahkan pembaca dalam memahami keterkaitan antara setiap komponen penelitian, mulai dari kondisi awal, analisis kebutuhan, pengembangan sistem, hingga hasil akhir yang ingin dicapai. Dengan adanya kerangka berpikir, penelitian menjadi lebih terarah dan terfokus, sehingga setiap langkah yang diambil dapat dipertanggungjawabkan secara ilmiah. Selain itu, kerangka ini juga menjadi pedoman bagi peneliti dalam menjalankan setiap tahapan penelitian secara konsisten, mulai dari identifikasi masalah hingga penarikan kesimpulan.



Gambar 2. 2 Kerangka Berpikir

1. Penjelasan Kerangka Berpikir

1. Identifikasi Masalah (Latar Belakang)

Tahap ini adalah titik awal yang menjelaskan mengapa sistem baru diperlukan. Masalah-masalah yang ditemukan di toko (misal: Martini) adalah:

1) Transaksi manual di buku nota rawan kesalahan.
2) Perhitungan pakai kalkulator lambat dan tidak terintegrasi.
3) Pengecekan stok fisik tidak efisien dan tidak real-time.
4) Laporan dijumlah manual rentan salah dan memakan waktu.
5) Nota mudah hilang/rusak tidak ada arsip digital.
6) Tidak ada hak akses semua orang bisa akses semua data, tidak aman.

2. Kondisi Awal

Bagian ini mempertegas keadaan nyata sebelum sistem dibuat, yaitu:

1) Semua proses masih manual.
2) Stok dicek secara fisik.
3) Laporan dikerjakan secara manual.
4) Masih menggunakan kalkulator untuk hitungan.

Kondisi awal ini menjadi pembanding untuk melihat efektivitas solusi nantinya.

3. Analisis Kebutuhan

Dari masalah dan kondisi awal, dilakukan analisis untuk menentukan:

1) Kebutuhan fungsional (fitur yang harus ada), misal: input transaksi, kelola stok, cetak nota, laporan otomatis, login pengguna.
2) Kebutuhan non-fungsional (kualitas sistem), yaitu:
   a. *User-friendly* mudah digunakan.
   b. *Cepat* respon waktu singkat.
   c. *Akurat* perhitungan otomatis dan minim error.
   d. *Aman* ada hak akses (role-based).

4. Metode *Prototype*

Ini adalah pendekatan pengembangan yang dipilih. Tahapannya:

1. Pengumpulan kebutuhan wawancara/observasi.
2. *Prototype* awal desain kasar tampilan dan alur.
3. Evaluasi umpan balik dari pengguna awal.
4. Mengkodekan sistem pembuatan aplikasi sesungguhnya.
5. Pengujian sistem uji coba fungsional dan non-fungsional.
6. Implementasi penerapan di toko.
7. Evaluasi sistem tinjauan setelah dipakai, apakah sesuai harapan.

Metode ini dipilih karena fleksibel dan memungkinkan perbaikan bertahap.

5. Hasil Penelitian

Kesimpulan akhir: Aplikasi POS (*Point of Sale*) siap digunakan di Toko Martini.

Artinya, semua tahapan di atas menghasilkan sebuah sistem yang dianggap mampu mengatasi masalah manual dan memenuhi kebutuhan toko.