BAB II.
TINJAUAN PUSTAKA

2.1 Teori Dasar

2.1.1 Rancang Bangun

Suatu proses yang diterapkan secara terstruktur yang bertujuan merancang, membuat, juga mewujudkan sistem, perangkat, atau aplikasi agar dapat memenuhi kebutuhan tertentu atau menyelesaikan suatu permasalahan merupakan definisi dari rancang bangun. Dalam bidang teknologi informasi, istilah ini umumnya mengacu pada tahapan perancangan dan pengembangan sistem berbasis teknologi, seperti *software*, aplikasi berbasis web, maupun sistem informasi yang terkomputerisasi. Menurut Pressman, rancang bangun perangkat lunak adalah pendekatan yang mencakup pengumpulan kebutuhan pengguna, perencanaan, perancangan, implementasi, pengujian, dan pemeliharaan sistem secara menyeluruh (Pressman, 2005).

Setyawan dan Hidayat menjelaskan bahwa rancang bangun mencakup proses iteratif yang menggabungkan aspek teknis dan kebutuhan pengguna untuk menghasilkan solusi yang optimal (Setyawan, E., & Hidayat, 2020). Tahapan dalam rancang bangun harus dilakukan secara kolaboratif dan melibatkan pemangku kepentingan guna memastikan solusi yang dihasilkan sesuai dengan kebutuhan operasional.

Rancang bangun berperan penting dalam memastikan pengembangan sistem dilakukan secara terstruktur, efisien, dan sesuai kebutuhan. Laudon dan Laudon menggarisbawahi bahwa rancang bangun yang baik dapat mengurangi risiko kegagalan proyek dan meningkatkan efisiensi operasional (Laudon, K. C., & Laudon, 2004).

2.1.2 Aplikasi

Sebuah objek berupa perangkat lunak terbentuk melalui proses perancangan dan pengembangan untuk membantu pengguna dalam menyelesaikan tugas atau aktivitas tertentu yakni pengertian dari aplikasi. Aplikasi telah menjadi elemen penting dalam berbagai aspek kehidupan, mulai dari dunia bisnis, pendidikan, hingga kebutuhan pribadi di era

digitalisasi ini. Fungsi utama aplikasi adalah untuk mengotomatisasi pekerjaan dan menyederhanakan proses sebelumnya secara manual berubah menjadi lebih praktis dan efisien.

Dalam konteks sistem informasi dan teknologi komputer, aplikasi dipahami sebagai kumpulan perintah atau instruksi logis yang dirancang secara sistematis agar dapat dijalankan oleh perangkat komputer dalam mengolah data masukan menjadi keluaran yang sesuai dengan tujuan pengguna. Konsep ini menjabarkan bahwa tidak hanya sekadar program aplikasi, tetapi juga merupakan bagian penting dari proses otomatisasi tugas, yang mampu membantu manusia dalam menyelesaikan pekerjaan secara lebih cepat dan akurat.

Secara umum, aplikasi dapat diartikan sebagai kumpulan instruksi yang terstruktur dan dijalankan oleh komputer untuk mengolah data masukan menjadi keluaran sesuai yang diharapkan. Pemahaman ini sejalan dengan pandangan Jogiyanto (2017), yang menyatakan bahwa aplikasi merupakan pemanfaatan komputer melalui sekumpulan instruksi-intruksi atau pernyataan disusun secara linear sehingga dapat memproses input dan merubahnya menjadi output. Sementara itu, menurut Safaat (2020), aplikasi adalah program yang telah siap digunakan dan dirancang untuk menjalankan fungsi tertentu bagi pengguna, termasuk mendukung penggunaan program lainnya sesuai kebutuhan pengguna. Definisi ini menekankan pada kemudahan penggunaan dan fungsionalitas langsung bagi pemakainya.

2.1.3 Web

Website merupakan sekumpulan halaman digital yang memuat berbagai jenis informasi, seperti tulisan, gambar, animasi, audio, hingga video, dan dapat diakses oleh *user* dengan bantuan koneksi internet dari berbagai belahan dunia. Abdulloh menjelaskan bahwa website adalah himpunan halaman yang menyajikan konten digital dalam berbagai format baik teks, gambar, suara, animasi, maupun kombinasi dari semuanya dan dapat diakses secara luas oleh publik melalui koneksi internet.

Perkembangan teknologi web terus mengalami kemajuan yang pesat, seiring dengan peningkatan kualitas infrastruktur jaringan internet.

Hal ini menjadikan website sebagai sarana yang sangat bermanfaat bagi pelaku industri, khususnya dalam menyediakan layanan tambahan dan menunjang aktivitas bisnis mereka secara lebih efektif dan efisien.

Dengan demikian, *website* berfungsi sebagai media yang memungkinkan distribusi informasi secara luas dan efisien, mendukung berbagai aktivitas seperti pendidikan, bisnis, dan hiburan. Perkembangan teknologi web yang pesat telah mendorong integrasi *website* dalam berbagai aspek kehidupan, menjadikannya komponen penting dalam era digital saat ini.

2.1.4 *Inventory*

Persediaan atau *inventory* merupakan elemen vital dalam operasional perusahaan, mencakup *raw material*, barang berproses, sampai produk jadi yang siap untuk didistribusikan atau dijual. Pengelolaan persediaan yang efektif bertujuan untuk memastikan ketersediaan barang sesuai kebutuhan, meminimalkan biaya penyimpanan, dan menghindari kekurangan atau kelebihan stok yang dapat mengganggu kelancaran operasional perusahaan.

Puteri menekankan bahwa manajemen persediaan yang efisien berperan krusial dalam distribusi barang dari pemasok ke pengguna akhir (Puteri et al., 2023). Gudang berfungsi sebagai tempat penyimpanan sementara sebelum barang didistribusikan, sehingga pengelolaan persediaan yang baik di gudang dapat meminimalkan risiko kekurangan atau kelebihan stok.

Implementasi sistem informasi manajemen persediaan barang dapat memberikan berbagai keuntungan, termasuk pengurangan biaya operasional, peningkatan kepuasan pelanggan, serta perbaikan dalam akurasi data dan pengambilan keputusan. Sistem ini memungkinkan perusahaan untuk meminimalkan risiko *overstock* dan *stockout*, yang keduanya dapat berdampak negatif terhadap profitabilitas dan reputasi bisnis.

2.1.5 Manajemen *Inventory*

Manajemen *Inventory* merupakan aktivitas yang melibatkan perencanaan, pengawasan, serta pengendalian terhadap arus barang dalam

sebuah sistem *inventory*. Kegiatan ini mencakup penerimaan, penyimpanan, pengambilan, dan distribusi barang, serta pemantauan dan pengelolaan stok. Manajemen *inventory* memiliki tujuan yaitu memastikan ketersediaan barang memenuhi kebutuhan, menjaga keakuratan data stok, serta meningkatkan efisiensi dalam operasional gudang.

Aspek penting dalam manajemen *inventory* modern adalah penerapan sistem penyimpanan yang efektif. Sutono menekankan pentingnya penerapan metode seperti *First In First Out* (FIFO), *Last In First Out* (LIFO), dan *First Expired First Out* (FEFO) yang disesuaikan terhadap karakteristik barang dan kebutuhan operasional (Sutono, 2019). Sistem ini tidak hanya memastikan pengelolaan inventori yang efisien tetapi juga menjaga kualitas barang yang disimpan.

Selain metode penyimpanan, pengaplikasian metode 5R (Ringkas, Rapi, Resik, Rawat, Rajin) dalam operasional gudang berperan penting dalam membentuk lingkungan kerja yang lebih baik lagi dalam hal efisiensi dan kedisiplinan. Prinsip ini mengutamakan kebersihan, keteraturan, serta kedisiplinan dalam aktivitas *inventory*, yang pada akhirnya mampu meningkatkan produktivitas sekaligus mengurangi potensi kesalahan dalam pengelolaan barang.

Dalam era digital, teknologi juga memainkan peran yang signifikan dalam meningkatkan efisiensi *inventory*. Salah satu teknologi yang banyak digunakan adalah sistem *inventory*. Sistem ini memungkinkan pemantauan pergerakan barang secara *real-time*, mengoptimalkan ruang penyimpanan, serta meningkatkan ketepatan pencatatan inventaris. Dengan adanya integrasi teknologi ini, proses *inventory* menjadi lebih sistematis, transparansi, serta memberikan kemudahan dalam perumusan keputusan.

Secara keseluruhan, penerapan teknik manajemen *inventory* yang tepat selain berdampak pada segi efisiensi operasional, namun juga dapat membantu kelancaran rantai pasok secara keseluruhan. Dengan pengelolaan gudang yang efektif, perusahaan dapat menekan biaya operasional, meningkatkan kepuasan pelanggan, serta mempertahankan daya saing di pasar.

2.1.6 *Sparepart*

Sparepart atau suku cadang adalah komponen yang digunakan untuk menggantikan bagian yang rusak atau aus pada suatu peralatan, mesin, atau kendaraan agar dapat berfungsi kembali seperti semula. Menurut Heizer & Render, *sparepart* adalah elemen penting dalam manajemen pemeliharaan dan operasi, yang berperan dalam memastikan keberlanjutan dan efisiensi peralatan atau kendaraan (Heizer, 2014). *Sparepart* bisa berupa bagian kecil seperti baut dan filter, hingga komponen utama seperti mesin dan transmisi.

Menurut Arnold et al., manajemen suku cadang menjadi salah satu aspek krusial dalam supply chain management, terutama dalam industri yang sangat bergantung pada kelangsungan operasional kendaraan atau mesin (Arnold, J. R., Chapman, S. N., & Clive, 1998). Penyimpanan dan pengelolaan *sparepart* yang baik dapat mengurangi waktu henti operasional dan meningkatkan produktivitas perusahaan.

Manajemen *sparepart* yang baik sangat penting dalam industri otomotif dan manufaktur, terutama untuk mengurangi downtime akibat kegagalan peralatan. Jika suatu perusahaan tidak memiliki stok *sparepart* yang cukup, maka perbaikan kendaraan atau mesin dapat tertunda, yang berakibat pada penurunan produktivitas dan efisiensi operasional.

2.1.7 Kendaraan

Kendaraan merupakan alat transportasi yang digunakan di jalan raya dan memiliki klasifikasi 2 macam, yaitu kendaraan bermotor dan tidak bermotor. Definisi dari kendaraan bermotor yaitu sebagai kendaraan yang menggunakan mesin sebagai penggerak utamanya, tidak termasuk kendaraan yang bergerak di atas rel.

Kadir (2006) menjelaskan bahwa kegiatan memindahkan orang atau barang dari satu lokasi ke lokasi lain dengan tujuan tertentu, menggunakan sarana yang digerakkan oleh manusia, hewan, atau mesin merupakan definisi dari transportasi. Sementara itu, menurut Sondakh et al. dalam Jurnal Sipil Statik (2019), kendaraan dapat dikelompokkan berdasarkan karakteristiknya menjadi beberapa jenis, seperti kendaraan berat, kendaraan ringan, sepeda motor, dan kendaraan tanpa mesin. Pengelompokan ini

sangat berguna dalam analisis lalu lintas serta perencanaan sistem transportasi yang lebih terstruktur dan efisien.

2.1.8 Database

Basis data atau dapat disebut sebagai *database*, merupakan komponen krusial dalam pengembangan aplikasi, termasuk aplikasi *inventory*. *Database* berfungsi memiliki fungsi yaitu sebagai fasilitas penyimpanan terstruktur untuk data yang dapat diakses dan dikelola dengan efisien. Dalam konteks aplikasi *inventory*, basis data menyimpan informasi terkait barang, seperti identitas produk, jumlah stok, lokasi penyimpanan, dan riwayat transaksi.

Menurut Dirgantara, *database* merupakan data-data yang memiliki relasi antar data dan disimpan secara menyatu dengan tingkat redundansi yang telah diatur sedemikian rupa, sehingga dapat mendukung satu hingga lebih banyak aplikasi secara efisien. Penyimpanan data dilakukan secara terpisah diaksesnya dari program, sementara proses penambahan, perubahan, dan pengambilan data dilakukan dengan pengelolaan yang terkendali (Dirgantara et al., 2023).

Dalam pengembangan aplikasi *inventory*, basis data memainkan peran yang sangat vital. Pemahaman mendalam tentang konsep dasar basis data, fungsi DBMS, serta tahapan perancangan basis data yang baik akan memastikan bahwa aplikasi yang dibangun mampu mengelola data dengan efisien dan mendukung kebutuhan operasional perusahaan secara optimal.

2.1.9 *Waterfall*

Metode *Waterfall* dalam pengembangan perangkat lunak merupakan pendekatan yang bersifat terstruktur dan berurutan, metode ini menerapkan sebelum melanjutkan ke tahap berikutnya harus menyelesaikan tahapan sebelumnya terlebih dahulu. Model ini juga dikenal sebagai model linier sekuensial atau model siklus hidup tradisional. Dalam pengembangan aplikasi *inventory*, penggunaan metode *Waterfall* memungkinkan proses pembangunan sistem dilakukan secara sistematis, sehingga setiap langkah dapat dirancang dan dijalankan dengan lebih teliti dan terorganisir.

Menurut Pressman, model *waterfall* adalah model klasik yang bersifat sistematis, berurutan dalam membangun *software*. Nama model ini

sebenarnya adalah "*Linear Sequential Model*" (Pressman, 2005). Lima langkah utama yang biasanya terdiri dari metodologi ini yaitu:

1. **Analisis Kebutuhan**: Tahapan awal para pengembang aplikasi dan pemangku kepentingan berkolaborasi untuk merumuskan kebutuhan dari perangkat lunak yang akan dibuat. Merumuskan persyaratan fungsional dan nonfungsional yang harus dipenuhi oleh perangkat lunak adalah tujuan utamanya.
2. **Perancangan**: Setelah persyaratan dikumpulkan, fase perancangan melibatkan merancang struktur dan arsitektur perangkat lunak secara keseluruhan. Fase ini mencakup merancang algoritma yang akan digunakan, basis data, komponen sistem, dan antarmuka pengguna.
3. **Implementasi**: Setelah perancangan selesai, tim pengembang mulai mengimplementasikan perangkat lunak sesuai dengan spesifikasinya. Semua komponen sistem dibuat, dan kode program ditulis.
4. **Pengujian**: Fase pengujian dilakukan setelah implementasi selesai hal ini dilakukan agar memastikan bahwa aplikasi sudah dikembangkan dapat berjalan dengan sesuai, memenuhi persyaratan, dan sesuai dengan harapan pengguna.
5. **Pemeliharaan**: Fase pemeliharaan dimulai setelah perangkat lunak diuji dan dirilis. Fase ini memiliki banyak kegiatan seperti perbaikan kesalahan, *update* fitur, dan penanganan masalah yang mungkin muncul setelah perangkat lunak digunakan secara aktif.

Metode *Waterfall* memberikan cara kerja yang runtut dalam proses pengembangan *software*, metode ini menerapkan bahwa setiap fase diselesaikan dengan teliti sebelum berlanjut ke fase berikutnya. Meskipun kurang fleksibel dalam menghadapi perubahan kebutuhan selama proyek berlangsung, pendekatan ini sangat sesuai untuk proyek-proyek yang spesifikasinya telah ditentukan dengan jelas sejak awal, seperti pada pembuatan aplikasi inventory.

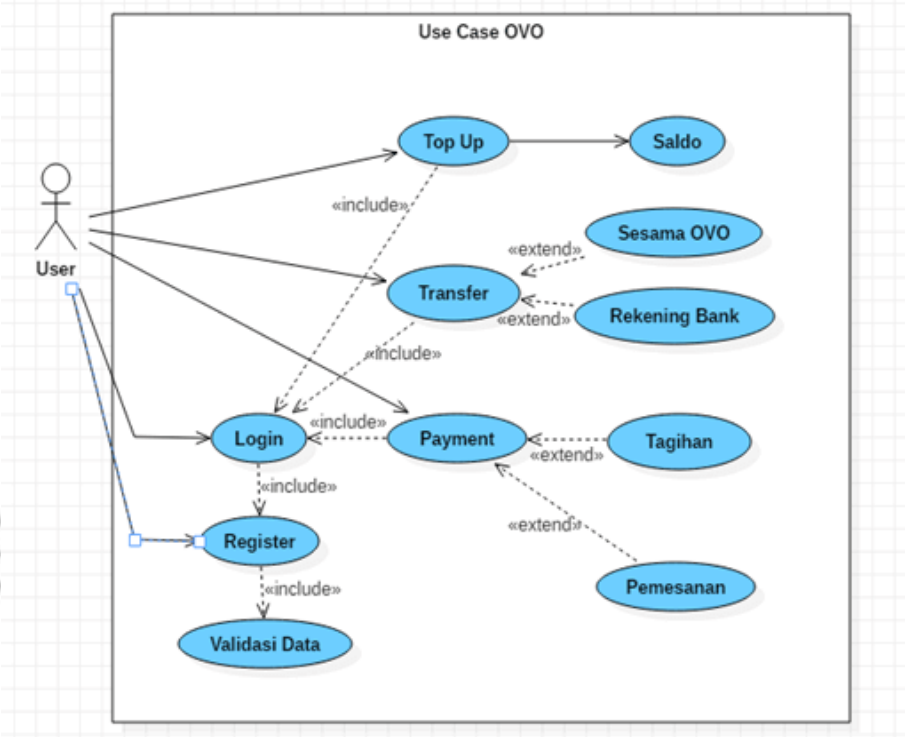
2.1.10 *Unified Modeling Language* (UML)

Standar bahasa pemodelan yang digunakan untuk mendefinisikan, menggambarkan, membangun, dan mencatat struktur sebuah sistem perangkat lunak merupakan definisi dari UML. UML menyajikan beragam jenis diagram yang saling terhubung, yang dapat diterapkan sebagai alat bantu bagi *developer* dalam merancang juga memahami sistem secara menyeluruh..

Grace Windu Gata mengungkapkan bahwa UML adalah bahasa spesifikasi yang bersifat standar dan dipakai untuk mendeskripsikan, mencatat, dan membangun sistem perangkat lunak. Selain itu, UML juga berperan sebagai metode dalam pengembangan sistem berbasis objek serta sebagai alat bantu dalam keseluruhan proses pengembangannya (Gata & Gata, 2013). Dalam implementasinya, para pengembang memanfaatkan empat jenis diagram utama UML untuk merancang perangkat lunak, yaitu:

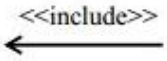
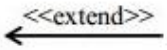
a) *Use Case Diagram*

Use Case Diagram berfungsi untuk memvisualisasikan relasi pengguna atau aktor terhadap sistem. Penggambaran diagram ini menjelaskan bagaimana pengguna memanfaatkan berbagai fitur atau layanan yang disediakan oleh sistem, sehingga memudahkan dalam mengidentifikasi kebutuhan fungsional dari perangkat lunak. Komponen dalam *Use Case Diagram* meliputi aktor yaitu perwakilan pengguna atau sistem luar, *use case* yang menggambarkan layanan atau fungsi sistem, serta garis penghubung yang menunjukkan relasi antara aktor dengan *use case* yang relevan.



Gambar 2.1 Contoh *Use Case Diagram*

Sumber: https://www.dicoding.com/blog/contoh-use-case-diagram/

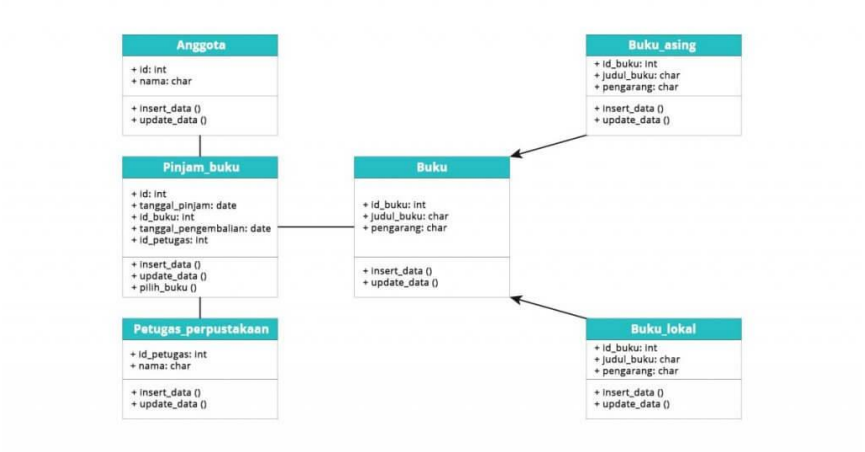
Simbol	Keterangan
	Aktor : Mewakili peran orang, sistem yang lain, atau alat ketika berkomunikasi dengan *use case*
	*Use case* : Abstraksi dan interaksi antara sistem dan aktor
	*Association* : Abstraksi dari penghubung antara aktor dengan use case
	*Generalisasi* : Menunjukkan spesialisasi aktor untuk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

Gambar 2.2 Simbol *Use Case Diagram*

Sumber: https://www.dicoding.com/blog/contoh-use-case-diagram/

b) *Class Diagram*

Class Diagram memiliki manfaat untuk menggambarkan kerangka internal dari sebuah sistem *software*. Diagram ini menjelaskan hubungan antar *class*, atribut-atribut yang menyusun sistem, fungsi atau metode yang dijalankan, serta interaksi antar objek yang terlibat di dalam sistem. Dalam pengembangan sistem berbasis objek, Class Diagram berperan penting karena membantu pengembang dalam merancang dan memahami struktur data serta keterkaitan antar komponen sistem secara lebih terstruktur.



Gambar 2.3 Contoh *Class Diagram*

Sumber: https://www.dicoding.com/blog/memahami-class-diagram-lebih-baik/

SIMBOL CLASS DIAGRAM

NO	GAMBAR	NAMA	KETERANGAN
1		*Generalization*	Hubungan dimana objek anak *(descendent)* berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (*ancestor*).
2		*Nary Association*	Upaya untuk menghindari asosiasi dengan lebih dari 2 objek.
3		*Class*	Himpunan dari objek-objek yang berbagi atribut serta operasi yang sama.
4		*Collaboration*	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu actor
5		*Realization*	Operasi yang benar-benar dilakukan oleh suatu objek.
6		*Dependency*	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri *(independent)* akan mempegaruhi elemen yang bergantung padanya elemen yang tidak mandiri
7		*Association*	Apa yang menghubungkan antara objek satu dengan objek lainnya

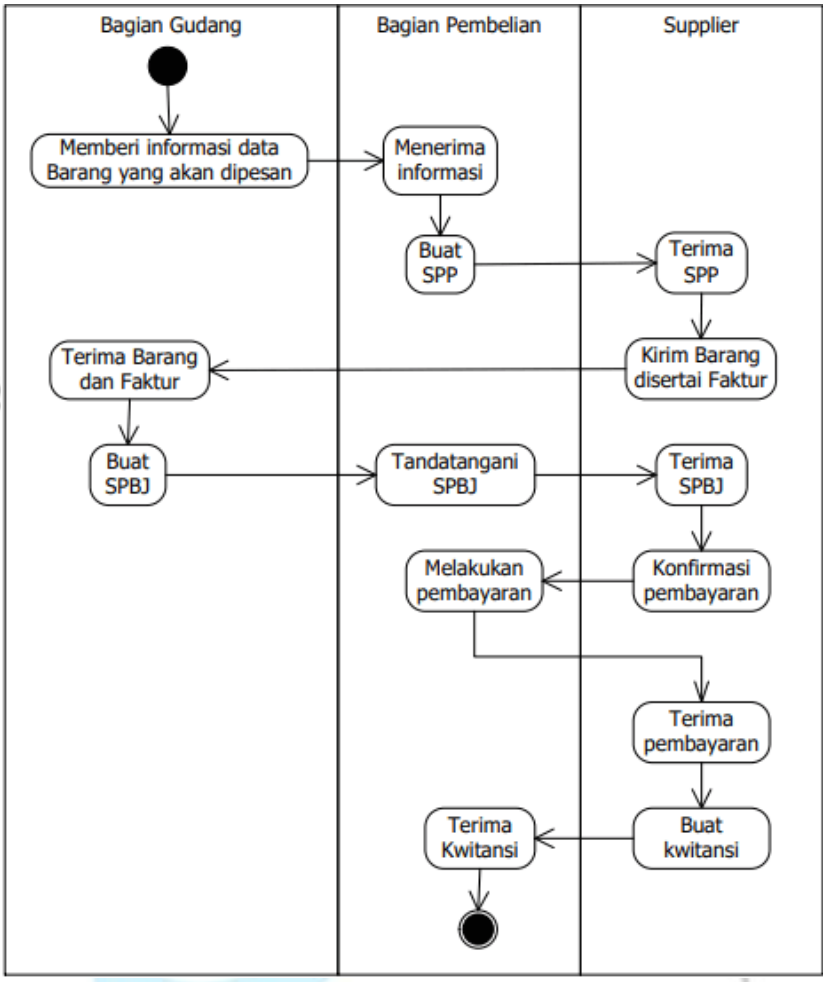
Gambar 2.4 Simbol *Class Diagram*

Sumber: https://medium.com/@sarahpramesthi/mengenal-jenis-jenis-structure-diagram-dan-behavior-diagram-dalam-diagram-uml-e597fc45355e

c) *Activity Diagram*

Activity Diagram adalah salah satu jenis diagram dalam *Unified Modeling Language* (UML) yang digunakan untuk menggambarkan alur kerja atau proses bisnis dalam suatu sistem. Diagram ini menunjukkan urutan langkah-langkah dalam sebuah aktivitas, termasuk

aliran data dan keputusan yang diambil dalam proses tersebut. *Activity Diagram* sering digunakan untuk memodelkan proses bisnis, logika pemrograman, serta interaksi pengguna dengan sistem secara visual.



Gambar 2.5 Contoh *Activity Diagram*

Sumber: https://www.dicoding.com/blog/apa-itu-activity-diagram/

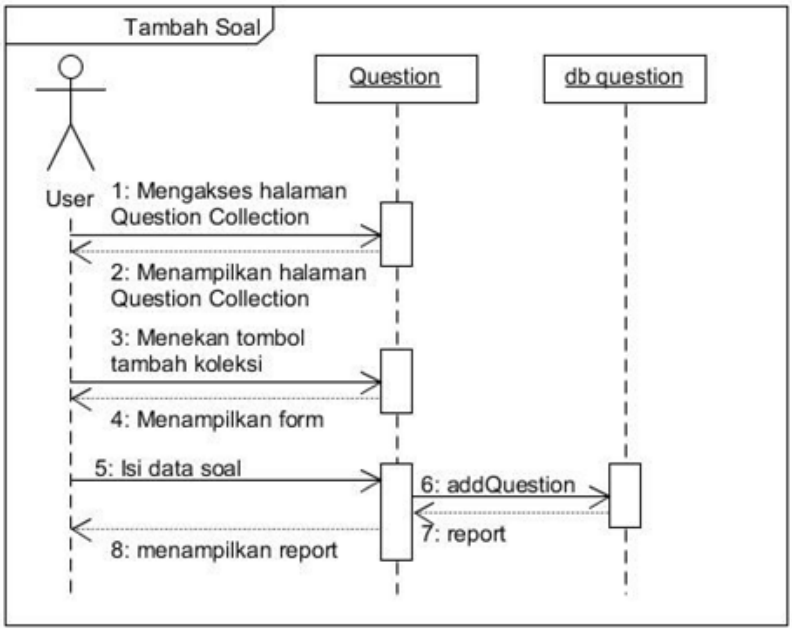
Simbol	Nama	Keterangan
	Status awal	Sebuah diagram aktivitas memiliki sebuah status awal.
	Aktivitas	Aktivitas yang dilakukan sistem,aktivitas biasanya diawali dengan kata kerja.
	Percabangan / Decision	Percabangan dimana ada pilihan aktivitas yang lebih dari satu.
	Penggabungan / Join	Penggabungan dimana yang mana lebih dari satu aktivitas lalu digabungkan jadi satu.
	Status Akhir	Status akhir yang dilakukan sistem, sebuah diagram aktivitas memiliki sebuah status akhir
	Swimlane	Swimlane memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi

Gambar 2.6 Simbol *Activity Diagram*

Sumber: https://www.dicoding.com/blog/apa-itu-activity-diagram/

d) *Sequence Diagram*

Sequence diagram berfungsi untuk mendefinisikan komunikasi antara para objek penyusun sistem dalam urutan waktu yang sudah ditentukan. Penggambaran pada diagram ini menjelaskan bagaimana objek-objek saling berkomunikasi, biasanya berbentuk pesan, selama eksekusi sebuah fungsi atau skenario tertentu. Pada diagram ini, objek digambarkan dalam bentuk kolom vertikal, dan pesan yang dikirim antar objek digambarkan dengan panah horizontal yang menunjukkan urutan interaksi. Urutan waktu digambarkan dari atas ke bawah, dengan pesan yang lebih tinggi menunjukkan interaksi yang terjadi lebih awal.



Gambar 2.7 Contoh *Sequence Diagram*

Sumber: https://medium.com/javanlabs/belajar-uml-activity-diagram-dan-sequence-diagram-4356c08ba137

Nama	Simbol	Fungsi
Object		Menggambarkan sebuah *class* atau *object*.
Activation boxes		Menggambarkan panjang waktu yang dibutuhkan sebuah *object* dalam mengerjakan tugasnya
Actors		Menggambarkan pengguna yang berinteraksi dengan sistem
Lifeline		Menggambarkan "garis hidup" sebuah *object*
Message		Menggambarkan pesan atau interaksi antar *object*
Message to Self		Menggambarkan pesan balikan atau reaksi dari *object* sebelumnya

Gambar 2.8 Simbol *Sequence Diagram*

Sumber: https://medium.com/javanlabs/belajar-uml-activity-diagram-dan-sequence-diagram-4356c08ba137

2.2 Tinjauan Studi

1. Jurnal "**Rancang Bangun Sistem Inventori Pengendalian Stok Barang Berbasis Java Pada PT Kalibesar Artah Perkasa**" karya Tuti Handayani, Ahmad Hapip Furqon, dan Supriyono (2020) dari Universitas Muhammadiyah Tangerang dan Universitas Muria Kudus,

mengkaji pengembangan sistem inventori berbasis Java. Pengumpulan data pada penelitian memanfaatkan teknik observasi, wawancara, dan kajian literatur. Pendekatan yang digunakan adalah *Object-Oriented Analysis* (OOA) dan *Object-Oriented Design* (OOD), dan memanfaatkan UML menjadi alat bantu pemodelan untuk mempermudah proses pengembangan.

2. Jurnal berjudul "**Rancang Bangun Sistem Informasi Inventory Barang (SINBAR) Berbasis Website**" ditulis oleh Agyztia Premana dari Universitas Muhadi Setiabudi Brebes (2019). Penelitian ini berfokus terhadap pembangunan sistem informasi inventaris barang *web-based* agar meningkatkan efisiensi pengelolaan stok di lingkungan kampus. Sistem ini mampu mencatat data barang masuk atau barang keluar, penyimpanan informasi stok bersifat *real-time*, dan menghasilkan *report* yang akurat serta cepat. Penggunaan sistem ini terbukti lebih efektif dibandingkan metode manual.
3. Jurnal "**Manajemen Gudang Sparepart PT Sinergining Adhi Selaras Menggunakan Metode 5S dan ABC**" oleh G. A. Putra (2020) dalam Jurnal Optimasi Teknik Industri membahas penerapan metode 5S dalam mengelola gudang sparepart. Penelitian ini menunjukkan bahwa dengan penerapan metode tersebut, kegiatan pergudangan menjadi lebih tertata, efisien, serta menumbuhkan kesadaran akan pentingnya pengelolaan gudang yang baik untuk menjaga kualitas barang.
4. Jurnal "**Rancang Bangun Sistem Inventory Gudang Menggunakan Metode *Waterfall* (Studi Kasus di CV. Aqualux Duspha Abadi Kudus Jawa Tengah)**" yang ditulis oleh E. Listiyan dan E. R. Subhiyakto dari Universitas Dian Nuswantoro Semarang (2021), membahas pengembangan sistem inventaris gudang dengan pendekatan *Waterfall*. Setiap tahapan, mulai dari analisis kebutuhan hingga pemeliharaan, dilakukan secara berurutan dan sistematis. Penelitian ini menunjukkan bahwa metode *Waterfall* mampu menghasilkan sistem yang efektif dan sesuai kebutuhan.

5. Jurnal "**Rancang Bangun Sistem Informasi Manajemen Gudang Apotek Menggunakan Model *Software Development Life Cycle* (SDLC) di Apotek MARIFA**" disusun oleh Abdul Muiz Hasyim, Yoyok Seby Dwanoko, dan Abdul Aziz dari Universitas Kanjuruhan Malang (2019). Jurnal ini membahas pengembangan sistem informasi gudang apotek untuk menggantikan pencatatan manual dengan sistem digital. Proses pengembangan dilakukan menggunakan model SDLC yang mencakup analisis kebutuhan, desain, implementasi, dan pengujian menggunakan *blackbox testing* dan UAT. Sistem ini bertujuan untuk mempermudah pengelolaan stok obat, pemantauan kedaluwarsa, serta pencatatan distribusi obat..