

BAB II

LANDASAN TEORI

2.1 Tinjauan Pustaka

Penelitian yang dilakukan oleh Ferdinan dan Fachrie (2024) berjudul "Pengembangan Aplikasi *Jasa Service* Peralatan Elektronik Berbasis *Android* Menggunakan *Payment Gateways*" bertujuan untuk mengembangkan aplikasi mobile yang memudahkan masyarakat dalam melakukan pemesanan jasa perbaikan peralatan elektronik tanpa harus datang langsung ke tempat *service*. Aplikasi tersebut dikembangkan berbasis *Android* dengan menyediakan fitur pemilihan kategori peralatan elektronik, pemilihan mitra teknisi, pemesanan layanan, riwayat pesanan, serta sistem pembayaran online yang terintegrasi dengan Midtrans sebagai *payment gateway*. Hasil penelitian menunjukkan bahwa aplikasi mampu mempermudah proses pemesanan jasa *service* elektronik, mempercepat proses transaksi, serta meningkatkan efisiensi pelayanan kepada pelanggan. Penelitian tersebut memiliki kesamaan, yaitu sama-sama mengembangkan aplikasi jasa *service* elektronik berbasis *Android* yang bertujuan mempermudah pelanggan dalam melakukan pemesanan layanan *service*. Adapun perbedaannya terletak pada teknologi dan fitur yang digunakan. Penelitian Ferdinan dan Fachrie berfokus pada integrasi sistem pembayaran menggunakan *Midtrans Payment Gateway*, sedangkan penelitian yang penulis lakukan menggunakan *Firebase Realtime Database* dan *Firebase Authentication* sebagai media pengelolaan data pengguna dan transaksi. Selain itu, penelitian ini menambahkan fitur pengelolaan status perbaikan, penentuan biaya *service* oleh karyawan, riwayat pesanan pelanggan, serta pengiriman informasi kepada pelanggan melalui WhatsApp sesuai dengan kebutuhan operasional Toko Prima Service Elektronik

Penelitian Akbar, dkk. (2021). Penelitian yang berjudul Aplikasi Online Berbasis Android "SI TekO" (Sistem Informasi Teknisi Online) Sebagai Solusi Mempermudah Masyarakat Dalam Mendapatkan Jasa *Service*. Pada era industry 4.0 ini masyarakat Indonesia tenaga manusia sudah digantikan oleh mesin atau computer maka dari itu peneliti membuat suatu aplikasi berbasis SI-TekO. Seperti halnya jika saat barang elektronika, IT, maupun instalasi rusak ingin memperbaiki

ataupun memasang alat dirumah akan menjadikan lebih mudah dengan menggunakan jasa online. Jasa sering dipandang sebagai suatu fenomena yang rumit. Kata jasa itu sendiri mempunyai banyak arti, dari mulai pelayanan pribadi sampai pelayanan jasa sebagai suatu produk. Produksi jasa mungkin berkaitan dengan produk fisik atau tidak. Sering sekali yang menjadi problematika adalah saat kita repot dalam bekerja sehingga kita kurang adanya waktu untuk mencari jasa reparasi perangkat elektronik kita, dan pasang instalasi yang kita harapkan. Maka dari itu peneliti membuat sebuah aplikasi berbasis android yang dinamakan SI-TekO. Hasil dari penelitian ini adalah aplikasi berbasis android yang SI-TekO.

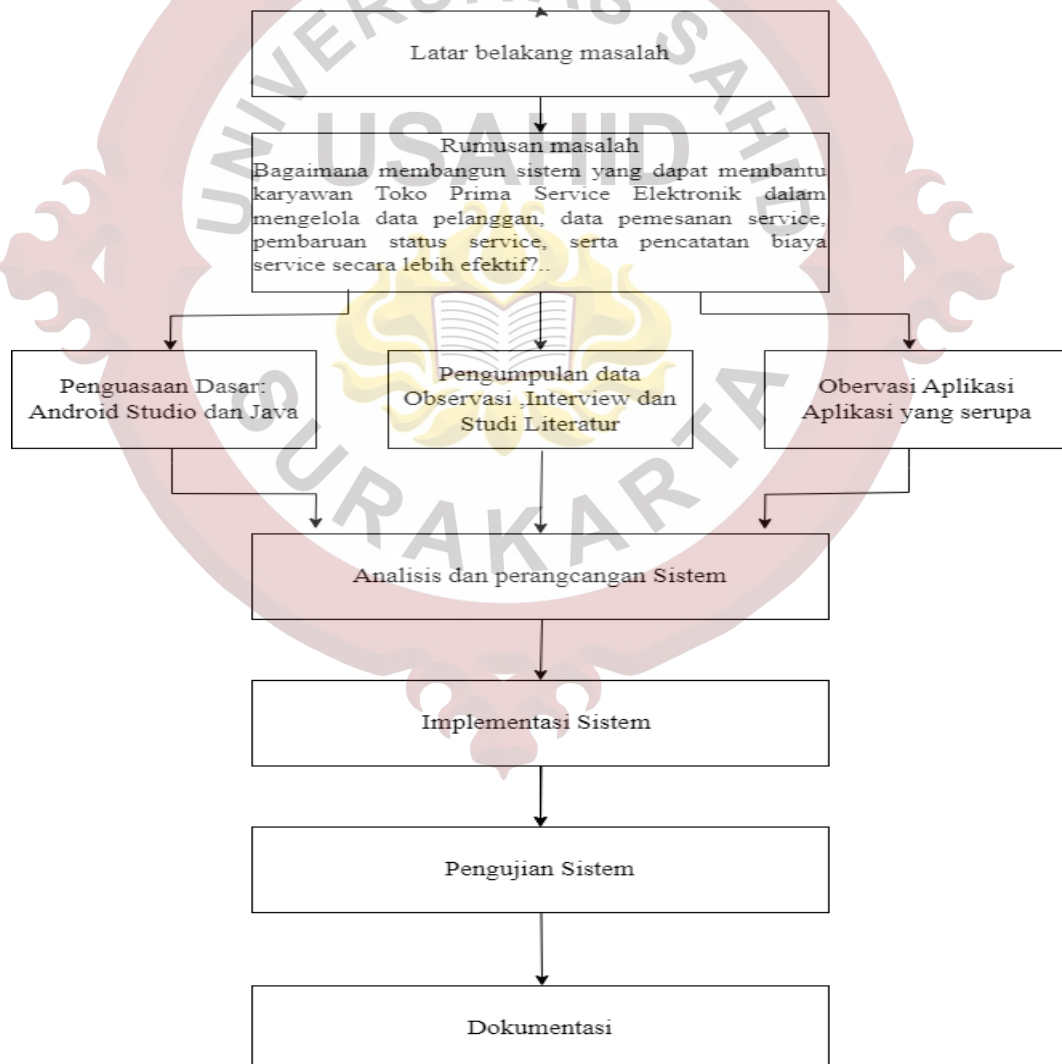
Penelitian yang dilakukan oleh Aulia, Suendri, dan Alda (2023) berjudul *"Build Android-Based Go Service Electronic Applications Using React Native Framework and Firebase Realtime Database"* bertujuan untuk mengembangkan aplikasi *Go Service Electronic* (GOSE) berbasis Android yang memudahkan masyarakat dalam mencari lokasi penyedia jasa service elektronik, mengetahui estimasi biaya perbaikan, melakukan konsultasi, serta mengajukan permintaan *service* secara daring. Aplikasi dikembangkan menggunakan *React Native Framework* sebagai *platform* pengembangan aplikasi mobile dan *Firebase Realtime Database* sebagai media penyimpanan data secara real-time. Penelitian ini memanfaatkan *Firebase Authentication* untuk autentikasi pengguna serta *Firebase Cloud Messaging* sebagai media notifikasi. Hasil penelitian menunjukkan bahwa aplikasi mampu mempermudah pelanggan memperoleh informasi layanan *service* elektronik, mengetahui estimasi biaya perbaikan, memperluas jangkauan pemasaran, serta meningkatkan efisiensi pelayanan. Persamaan penelitian ini dengan penelitian penulis adalah sama-sama mengembangkan aplikasi jasa *service* elektronik berbasis Android menggunakan *Firebase Realtime Database* dan bertujuan mempermudah pelanggan memperoleh layanan secara digital. Perbedaannya, penelitian Aulia, Suendri, dan Alda menggunakan *React Native* sehingga aplikasi dapat berjalan pada Android dan iOS, sedangkan penelitian ini menggunakan *Android Studio* dengan bahasa pemrograman Java yang dikhususkan untuk Android. Selain itu, penelitian ini menambahkan fitur pengelolaan status

service, penentuan biaya perbaikan, riwayat pesanan pelanggan, dan integrasi WhatsApp sesuai kebutuhan operasional Toko Prima Service Elektronik.

Berdasarkan penelitian terdahulu, pengembangan aplikasi pelayanan jasa berbasis Android umumnya masih berfokus pada pemesanan layanan dan pengelolaan data pelanggan. Penelitian ini mengembangkan aplikasi jasa *service* elektronik dengan fitur pengelolaan *status service*, nota PDF, berbagi nota, laporan transaksi berdasarkan periode, serta pengelolaan data secara *real-time* menggunakan *Firestore Realtime Database*.

2.2 Kerangka Pemikiran

Gambar dan Penjelasan dari kerangka pemikiran dapat disajikan pada Gambar 2.1



Gambar 2.1 Struktur Kerangka Pemikiran

1. Latar Belakang masalah

Permasalahan yang dihadapi Toko Prima Service Elektronik adalah proses pendataan pelanggan, pemesanan *service*, dan pemantauan status perbaikan yang masih dilakukan secara manual melalui telepon dan WhatsApp. Hal tersebut menyebabkan pengelolaan data kurang efektif, meningkatkan risiko kesalahan pencatatan, serta menyulitkan karyawan dalam memberikan pelayanan kepada pelanggan.

2. Rumusan Masalah

Bagaimana membangun sistem yang dapat membantu karyawan Toko Prima Service Elektronik dalam mengelola data pelanggan, data pemesanan *service*, pembaruan *status service*, serta pencatatan biaya *service* secara lebih efektif?

3. Pengumpulan Data

Pengumpulan data pada penelitian ini ada tiga yaitu *Observasi*, *interview* dan *Studi literatur*. Penelitian ini mengumpulkan semua data yang di butuhkan dengan melakukan pengamatan dari beberapa jurnal yang serupa.

4. Penguasaan Dasar

Penguasaan dasar dalam penelitian ini adalah penguasaan yang telah di ketahui atau dikuasai penulis mengenai *Android studio* dan *java, javascript*.

5. Observasi Aplikasi

Observasi dalam penelitian ini adalah penguasaan terhadap *tools* atau spesifikasi yang akan digunakan untuk membuat aplikasi *mobile* jasa pemesanan *service*.

6. Analisis dan Perancangan sistem

Analisis dan perancangan sistem dalam penelitian ini menggunakan UML (*Unified Modelling Language*) dengan menggunakan *use case diagram*, *class diagram*, *sequence diagram*, *activity diagram*, *deployment diagram*, *component diagram*. agar mempermudah dalam menganalisis dan merancang sebuah sistem yang baru.

7. Implementasi sistem

Setelah dilakukan analisis dan perancangan kemudian dilakukan sebuah implementasi sistem dalam penelitian ini adalah sistem dari proses pembuatan aplikasi jasa *service* elektronik *mobile*.

8. Pengujian Sistem

Pengujian sistem yang di lakukan yaitu dengan *Blackbox testing* dan juga *Compability testing*, *Blackbox testing* merupakan pengujian pada jalan nya fungsi pada aplikasi yang di uji. *Compability* merupakan cara untuk mengetahui apakah aplikasi yang telah dibuat mudah digunakan dengan mencobanya pada *user*.

User diminta untuk menyelesaikan sebuah tugas, biasanya saat mereka diawasi oleh peneliti, untuk melihat dimana mereka menemui masalah dan kebingungan. Jika lebih banyak orang mengalami masalah yang sama, rekomendasi akan dibuat untuk mengatasi permasalahan dalam penggunaan tersebut. mencoba ke beberapa konsumen atau ke beberapa *device android*.

9. Dokumentasi

Dokumenstasi dari perancangan awal hingga ke hasil dan di masukan kedalam sebuah laporan.

2.3 Teori Pendukung

Berikut beberapa landasan teori yang digunakan dalam penelitian ini:

2.3.1 Aplikasi Mobile

Aplikasi mobile merupakan sebuah aplikasi yang telah terintegrasi (IDE) yang disediakan oleh *Google* untuk membangun aplikasi Android. Yang mana IDE sendiri merupakan sebuah aplikasi yang berguna untuk mengembangkan aplikasi android, menyediakan berbagai alat dan fitur yang mempermudah proses pengembangan. Dengan *Android Studio*, pengembang dapat membuat aplikasi Android dengan antarmuka pengguna yang menarik dan fungsionalitas yang kaya (Zalukhu and Pangaribuan 2023). *Mobile application* juga memiliki beberapa manfaat, contohnya perusahaan barang atau jasa menggunakan *mobile application* untuk mengenalkan hasil produksi barang atau jasanya kepada pelanggan agar lebih tertarik memilih barang atau jasa perusahaan tersebut.

2.3.2 *Android*

Sistem operasi *Android* merupakan sebuah *platform* perangkat *mobile* berbasis Linux yang dirancang untuk menyediakan pengalaman pengguna yang optimal melalui kombinasi antara *middleware*, sistem operasi, dan aplikasi. Sebagai *platform* terbuka, *Android* memberikan kebebasan yang luas kepada para pengembang untuk menciptakan berbagai aplikasi inovatif yang dapat berjalan di dalam ekosistemnya. Pendekatan ini tidak hanya mendorong kreativitas dan inovasi, tetapi juga memungkinkan pengembangan solusi yang beragam sesuai dengan kebutuhan pengguna di berbagai bidang (Sholeh dkk., 2022).

2.3.3 *UI/UX*

User Interface dan *User Experience* (UI/UX) memiliki peranan penting dalam pembuatan sebuah aplikasi, karena desain pada sebuah aplikasi harus rapi dan terorganisir sehingga pengguna dapat dengan mudah memakai fitur-fitur yang telah disediakan oleh sebuah aplikasi. Desain *User Interface* dan *User Experience* (UI/UX) juga harus sesuai dengan kebutuhan pengguna dari aplikasi yang akan dibangun mulai dari desain tampilan, fitur-fitur, dan berbagai kebutuhan lainnya. (Solikin et al., 2022).

2.3.4 *ReactNative*

Menurut Wibawa dan Naufal (2023), *React Native* merupakan *framework JavaScript* yang dikembangkan oleh Facebook untuk membangun aplikasi mobile pada sistem operasi *Android* dan *iOS*. *Framework* ini memungkinkan pengembang membuat aplikasi lintas *platform* (*cross-platform*) menggunakan satu basis kode (*single codebase*), sehingga proses pengembangan menjadi lebih efisien serta dapat mengurangi waktu dan biaya pengembangan aplikasi.

2.3.5 *Layanan/Jasa*

Menurut Damayanti dan Haryono (2021), jasa merupakan suatu aktivitas atau layanan yang diberikan oleh penyedia jasa kepada pelanggan untuk memenuhi kebutuhan dan menciptakan nilai bagi pelanggan. Dalam bidang jasa, kualitas layanan menjadi faktor utama yang menentukan kepuasan pelanggan sehingga penyedia jasa perlu merancang layanan yang efektif, baik secara konvensional maupun elektronik.

Selain itu, jasa merupakan setiap tindakan atau aktivitas yang dapat ditawarkan oleh satu pihak kepada pihak lain yang pada dasarnya tidak berwujud (intangible) dan tidak mengakibatkan perpindahan kepemilikan. Produksi jasa dapat berkaitan maupun tidak berkaitan dengan suatu produk fisik. Definisi tersebut menjelaskan bahwa jasa lebih menekankan pada pemberian manfaat, pengalaman, dan kepuasan kepada pelanggan dibandingkan dengan penyerahan suatu barang.

2.3.6 *Software Development Kit (SDK)*

Menurut Regita (2023), Android SDK (*Software Development Kit*) merupakan seperangkat alat pengembangan (*development tools*) yang digunakan untuk membangun aplikasi pada *platform Android*. Android SDK menyediakan berbagai komponen yang diperlukan dalam proses pengembangan aplikasi, seperti pustaka (*libraries*), *Application Programming Interface (API)*, *emulator*, *debugger*, dokumentasi, serta alat bantu lainnya sehingga proses pembuatan, pengujian, dan implementasi aplikasi Android dapat dilakukan dengan lebih efektif dan efisien.

2.3.7 *Perbaikan/Service*

Menurut Ferdinan dan Fachrie (2024), *service* atau perbaikan merupakan layanan yang diberikan kepada pelanggan untuk mengatasi kerusakan pada peralatan elektronik agar perangkat dapat kembali berfungsi dengan baik. Layanan *service* mencakup proses identifikasi kerusakan, perbaikan komponen, penggantian suku cadang apabila diperlukan, serta penyelesaian transaksi pelayanan kepada pelanggan. Seiring perkembangan teknologi, proses pelayanan *service* dapat dilakukan melalui aplikasi berbasis Android sehingga pelanggan dapat melakukan pemesanan layanan secara daring tanpa harus datang langsung ke tempat *service*.

2.3.8 *Firebase*

Layanan ini dirancang untuk memudahkan pengembang aplikasi *mobile* dan *web* dengan menyediakan infrastruktur yang dibutuhkan untuk pengelolaan *backend*, sehingga memungkinkan pengembang fokus pada pengembangan logika aplikasi dan fitur utama. *Firebase* didirikan pada tahun 2011 oleh Andrew Lee dan James Tamplin dengan tujuan menyederhanakan proses pengembangan aplikasi (Andilala dkk., 2021).

2.3.9 Android Studio

Menurut penelitian yang dipublikasikan pada SISFO: Jurnal Ilmiah Sistem Informasi (2024), Android Studio merupakan *Integrated Development Environment (IDE)* yang dikembangkan oleh Google untuk pengembangan aplikasi Android. Android Studio dibangun di atas platform IntelliJ IDEA dan menyediakan berbagai fitur yang mendukung proses perancangan, penulisan kode, pengujian, debugging, serta implementasi aplikasi Android secara lebih efektif. Selain itu, Android Studio telah menggantikan *Eclipse Android Development Tools (ADT)* sebagai IDE resmi dalam pengembangan aplikasi Android.

2.3.10 Java

Menurut Sanjaya, J. J., & Susilo, J. (2024). Java merupakan bahasa pemrograman berorientasi objek (*Object-Oriented Programming/OOP*) yang banyak digunakan dalam pengembangan berbagai jenis perangkat lunak, termasuk aplikasi *desktop*, *web*, dan *Android*. Java memiliki karakteristik *Write Once, Run Anywhere (WORA)*, yaitu program yang ditulis satu kali dapat dijalankan pada berbagai platform yang mendukung *Java Virtual Machine (JVM)* tanpa memerlukan perubahan pada kode program. Selain itu, Java dikenal memiliki stabilitas yang baik, ekosistem yang luas, serta dukungan pustaka (*library*) yang lengkap sehingga masih menjadi salah satu bahasa pemrograman yang banyak digunakan dalam pengembangan aplikasi Android.

2.3.11 Whimsical

Jane Doe (2021), seorang peneliti di bidang desain interaksi, dalam studinya "*The Impact of Collaborative Tools on UI/UX Design Efficiency*," menekankan bahwa *Whimsical* memungkinkan tim desain untuk berkomunikasi dan bekerja sama dengan lebih baik, meningkatkan efisiensi dan kualitas hasil desain mereka. Dalam *website* resmi *Whimsical* dijelaskan bahwa *Whimsical* merupakan aplikasi berbasis web yang memiliki kemampuan atau fungsionalitas untuk membuat *flowchart*, *wireframe*, *flowchart*, dan lain-lain. Aplikasi ini gratis untuk diakses dan ada juga paket premium untuk lebih banyak fitur.

2.3.12 *Black Box Testing*

Black Box Testing merupakan metode pengujian perangkat lunak yang berfokus pada pengujian fungsi sistem berdasarkan masukan (*input*) dan keluaran (*output*) tanpa memperhatikan struktur kode program di dalamnya. Pengujian ini bertujuan untuk memastikan bahwa setiap fitur atau fungsi aplikasi telah berjalan sesuai dengan kebutuhan pengguna dan spesifikasi sistem. *Metode Black Box Testing*, penguji hanya mengevaluasi apakah sistem menghasilkan keluaran yang benar berdasarkan skenario pengujian yang telah ditentukan. Frayudha, A. D., Pande, I. R., & Juwita, M. B. (2024).

2.3.13 *Compatibility Testing*

Compatibility Testing merupakan salah satu jenis pengujian perangkat lunak yang dilakukan untuk memastikan bahwa aplikasi dapat berjalan dengan baik pada berbagai lingkungan yang berbeda, seperti versi sistem operasi, ukuran layar, jenis perangkat, resolusi, maupun spesifikasi perangkat keras. Pengujian kompatibilitas bertujuan untuk mengetahui apakah aplikasi tetap berfungsi secara optimal dan memberikan pengalaman pengguna yang konsisten ketika dijalankan pada berbagai perangkat Android yang memiliki karakteristik berbeda. Jihaadi, F. I. H., Hozairi, & Hoiriyah. (2024).

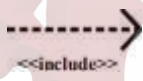
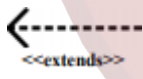
2.3.14 *UML(Unified Modeling Language)*

Menurut Menurut Kroenke dan McKinney (2021), *Unified Modeling Language* (UML) merupakan sekumpulan diagram, struktur, dan teknik yang digunakan untuk memodelkan serta merancang perangkat lunak berbasis objek. UML adalah bahasa pemodelan standar yang digunakan untuk memvisualisasikan, menspesifikasikan, mendokumentasikan, dan membangun sistem perangkat lunak. Pada UML terdapat beberapa diagram di antaranya sebagai berikut:

a. *Use Case Diagram*

Menurut Masgo & Santoso (2022), *Use Case Diagram* adalah diagram yang menggambarkan hubungan dan interaksi antara aktor dengan sistem serta digunakan untuk mengetahui fungsi-fungsi yang terdapat dalam sistem. Simbol-simbol *Use Case Diagram* dapat dilihat pada Tabel 2.1.

Tabel 2.1 *Use Case Diagram*

Simbol	Nama	Keterangan
	<i>Actor</i>	<i>Actor</i> tersebut mempresentasikan seseorang atau sesuatu (seperti perangkat, sistem lain) yang berinteraksi dengan sistem.
	<i>Use CaSe</i>	Gambaran fungsionalitas dari suatu sistem, sehingga <i>customer</i> atau pengguna sistem paham dan mengerti mengenai kegunaan sistem yang akan dibangun.
	<i>Association</i>	Jalur komunikasi antar actor dengan use case yang saling berpartisipasi.
	<i>Include</i>	Kejadian yang harus terpenuhi agar sebuah <i>event</i> dapat terjadi, di mana pada kondisi ini sebuah <i>use case</i> adalah bagian dari <i>use case</i> lainnya.
	<i>Extend</i>	Perluasan dari <i>use case</i> lain jika kondisi atau syarat terpenuhi.
	<i>Generalizaon</i>	Hubungan antara use case umum dengan use case yang lebih spesifik, yang mewarisi dan menambah fitur terhadapnya.

b. *Activity Diagram*

Menurut (Julianto & Setiawan, 2021) Diagram aktivitas atau *activity diagram* menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis atau menu yang ada pada perangkat lunak. Simbol – simbol *Activity Diagram*. *Activity diagram* juga digunakan untuk menggambarkan interaksi antara beberapa *use case*. Simbol-simbol yang digunakan dalam *activity diagram* dapat dilihat pada Tabel 2.2

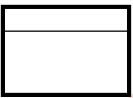
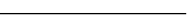
Tabel 2.2 *Activity Diagram*

Simbol	Nama	Keterangan
	Status Awal (<i>Initial</i>)	Status awal aktivitas sistem, sebuah diagram aktivitas memiliki sebuah status awal.
	Aktivitas	Aktivitas yang dilakukan sistem, aktivitas biasanya diawali dengan kata kerja.
	Percabangan (<i>Decision</i>)	Percabangan dimana ada pilihan aktivitas yang lebih dari satu.
	Penggabungan (<i>Join</i>)	Asosiasi penggabungan dimana lebih dari satu aktivitas lebih dari satu.
	Status Akhir (<i>Final</i>)	Status akhir yang dilakukan sistem, sebuah diagram aktivitas memiliki status akhir.
	<i>Swimlane</i>	Memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi.

c. *Class Diagram*

Menurut (Wira et al., 2021) *Class diagram* ialah menjelaskan secara garis besar mengenai kelas-kelas perancangan sistem dari sudut pandang struktur sistem yang dapat memperjelas fungsi - fungsinya. *Class diagram* merupakan hubungan antar kelas dan penjelasan detail tiap-tiap kelas di dalam model desain dari suatu sistem, juga memperlihatkan aturan-aturan dan tanggung jawab entitas yang menentukan perilaku sistem. *Class diagram* juga menunjukkan atribut- atribut dan operasi- operasi dari sebuah kelas dan *constraint* yang berhubungan dengan objek yang dikoneksikan. Pada *class diagram* terdapat beberapa simbol dan beberapa cara penulisan, seperti pada Tabel 2.3.

Tabel 2.3 *Class diagram*

Simbol	Nama	Keterangan
	<i>Generalization</i>	Hubungan dimana objek anak (<i>despended</i>) berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (<i>ansector</i>).
	<i>Nary Association</i>	Upaya untuk menghindari asosiasi dengan lebih dari 2 objek.
	<i>Class</i>	Himpunan dari objek-objek yang berbagi atribut serta operasi yang sama.
	<i>Collaboration</i>	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu <i>actor</i> .
	<i>Realization</i>	Operasi yang benar-benar dilakukan oleh suatu objek.
	<i>Dependency</i>	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri (<i>independent</i>) akan mempengaruhi elemen yang tidak mandiri.
	<i>Association</i>	Apa yang menghubungkan antara objek satu dengan objek lainnya.

d. *Sequence Diagram*

Sequence Diagram adalah jenis diagram UML (*Unified Modeling Language*) yang digunakan untuk memodelkan interaksi antar objek dalam urutan waktu tertentu. Diagram ini menampilkan bagaimana objek berkomunikasi melalui pesan (*messages*) dalam sebuah skenario atau proses. Pada *Sequence Diagram* terdapat beberapa simbol dan beberapa cara penulisan seperti Tabel 2.4

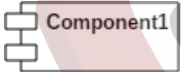
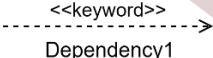
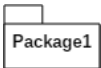
Tabel 2.4 *Sequence Diagram*

Simbol	Nama	Deskripsi
	Actor	Entitas eksternal yang berinteraksi dengan sistem.
	Lifeline	Garis vertikal yang menunjukkan eksistensi objek selama proses berlangsung.
	Message	Garis panah yang menunjukkan komunikasi antara objek.
	Activation bar	Menunjukkan saat objek sedang aktif menjalankan proses.

e. *Component Diagram*

Component Diagram adalah diagram UML yang digunakan untuk memvisualisasikan struktur fisik dari sistem, yang menunjukkan bagaimana komponen-komponen perangkat lunak saling berhubungan dapat disajikan pada Tabel 2.5

Tabel 2.5 *Component Diagram*

Simbol	Nama	Deskripsi
	Component	Bagian dari sistem yang memiliki fungsi tertentu.
	Interface	Menunjukkan titik akses antar komponen.
	Dependency	Hubungan yang menunjukkan bahwa satu komponen bergantung pada komponen lain.
	Node	Representasi fisik dari lingkungan di mana komponen berjalan.
	Package	Merupakan sebuah bungkus dari satu atau lebih komponen
	Link	Digunakan untuk mengarahkan relasi antar komponen

f. *Deployment Diagram*

Deployment Diagram adalah diagram UML yang digunakan untuk memvisualisasikan arsitektur fisik dari sistem perangkat lunak. Diagram ini menunjukkan bagaimana komponen perangkat lunak diinstal pada perangkat keras dapat disajikan pada Tabel 2.6

Tabel 2.6 *Deployment Diagram*

Simbol	Nama	Deskripsi
	Node	Representasi perangkat keras seperti server atau client.
	Artifact	Produk akhir dari proses pembangunan perangkat lunak seperti file executable.
	Deployment	Hubungan antara node dan artifact yang menunjukkan proses penempatan.