BAB II
TINJAUAN PUSTAKA

Dalam bab dua, peneliti akan membahas secara rinci tinjauan pustaka yang relevan dengan topik penelitian sebagai referensi dan perbandingan terhadap beberapa teori yang terkait dalam tugas akhir berjudul "Pengembangan Aplikasi Pendaftaran Produk Berbasis Mobile Untuk Meningkatkan Performa Di PT XYZ ". Penelitian ini bertujuan untuk mengembangkan aplikasi mobile, sehingga teori dasar yang akan dijelaskan mencakup konsep pengembangan aplikasi, dan sistem database yang akan digunakan.

2.1 Teori Dasar

2.1.1 Definisi Pengembangan Aplikasi

Mengacu kepada penelitian yang dilakukan oleh Tim Penyusun Kamus Pusat Pembinaan dan Pengembangan Bahasa (1989: 414), pengertian etimologis pengembangan dapat ditelusuri dari kata "kembang" yang mencerminkan peningkatan yang lebih baik dalam hal pribadi, pemikiran, pengetahuan, dan sejenisnya. Pengembangan merujuk pada proses, metode, dan tindakan yang terlibat. Selain itu, pandangan Burhan Nurgiyanto (2008:13) menyatakan bahwa pengembangan diartikan sebagai kegiatan menyusun, melaksanakan, mengevaluasi, dan memperbaiki suatu aktivitas.

Sesuai dengan penjelasan Al-Bahra Bin Ladjmudin (2005: 3-6), perkembangan sistem dapat didefinisikan sebagai tindakan untuk menata sistem baru yang menggantikan serta mengembangkan kembali sistem yang sudah ada. Sistem lama perlu disesuaikan atau diganti karena adanya kebutuhan untuk menangani masalah yang muncul, menjawab instruksi yang disajikan, atau mengambil peluang yang tersedia. Dengan kemunculan

pengembangan sistem baru, diharapkan terjadi peningkatan dalam beberapa hal seperti berikut:[1]

a. Peningkatan kualitas penyajian informasi.

b. Peningkatan hasil kerja sistem agar lebih efektif.

c. Optimalisasi efisiensi operasional. [1]

Sesuai dengan pendapat yang diungkapkan oleh para ahli di atas, dapat disimpulkan bahwa pengembangan melibatkan kegiatan yang terarah dan melibatkan perencanaan komprehensif untuk pembuatan atau perbaikan sebuah sistem yang disesuaikan agar memenuhi kebutuhan fungsional yang diinginkan dan memiliki tingkat kualitas yang lebih baik. Perkembangan dilakukan untuk melakukan perkembangan-perkembangan yang diperlukan terhadap hasil yang ingin dicapai.

Pengembangan aplikasi dapat diartikan sebagai serangkaian kegiatan dalam lingkup ilmu pengetahuan dan teknologi melibatkan tahap perancangan serta pembuatan pernyataan atau instruksi pada komputer untuk mengubah masukan menjadi keluaran dengan tujuan meningkatkan manfaat, fungsi, dan pengetahuan dalam teknologi yang ada saat ini untuk menciptakan teknologi baru. [2]

Menurut Sudarsono, dkk (2013: 186), *Research and Development* (R&D) merujuk pada riset pengembangan teknologi. Untuk menghasilkan produk atau sistem yang sesuai standar dengan integrasi yang terkoordinasi dan sistematis, para pengembang memerlukan pendekatan terstruktur pengembangan sistem. Dalam buku Ian Sommerville (2015: 19), rekayasa perangkat lunak merupakan keterampilan kerja di mana perangkat lunak

[1] Kamus Besar Bahasa Indonesia. Tim Penyusun Kamus Pusat Pembinaan dan Pengembangan Bahasa. (1989). Halaman 414.
[2] Nurgiyanto, B. (2008). Mengembangkan Kreativitas Anak dengan Metode Cerita. Halaman 13.

yang dirancang khusus, baik untuk penyematkan pada perangkat atau sebagai produk perangkat lunak seperti sistem informasi. [3]

Semua pengembang perangkat lunak, tanpa terkecuali, perlu melalui serangkaian tahapan selama proses pengembangan perangkat lunak. Tahapan-tahapan tersebut meliputi siklus persyaratan, analisis, desain, implementasi, dan pengujian. Selain itu, dalam buku Roger S. Pressman (2010: 31) disebutkan bahwa pemantauan dan pengawasan proyek, manajemen risiko, pengamanan kualitas, manajemen konfigurasi, evaluasi teknis, dan elemen lainnya diterapkan pada setiap langkah proses pengembangan. [4]

Seluruh tahap dalam pengembangan perangkat lunak dapat mengikuti urutan tersebut, tetapi masing-masing tahap memiliki fokus yang berbeda, namun masing-masing tahap memiliki penekanan yang berbeda.[5] Dalam konteks ini, penting untuk menentukan urutan tahapan yang sesuai dengan model pengembangan yang digunakan. Model pengembangan ini juga dikenal dengan istilah *Software Development Life Cycle* (SDLC), yang sering digunakan dan mencakup beberapa pendekatan seperti *waterfall, iterative, iterative and incremental, evolutionary prototyping, dan ad-hoc* atau *code-and-fix* SDLC. [6]

2.1.2 Pengertian Analisis dan Desain Sistem

Menurut Mulyani (2017), analisis dan perancangan sistem dapat diartikan sebagai proses penentuan langkah-langkah yang diperlukan untuk memenuhi kebutuhan data sistem terbaru. Orientasi dari perancangan sistem ini adalah untuk memenuhi kebutuhan pengguna sistem (user systems requirement) serta memberikan pemaparan yang detail mengenai rancangan yang komprehensif.

Muharto (2016) mengemukakan bahwa analisis dan desain sistem merupakan tahap di mana keterampilan perancangan diperlukan untuk

komponen-komponen komputer yang akan digunakan dalam sistem. Hal ini meliputi penentuan pilihan perangkat keras dan perangkat lunak untuk sistem yang baru.

Whitten & Bentley (2007) menjelaskan bahwa dalam kegiatan analisis dan desain sistem terdapat beberapa pendekatan masalah yang digunakan. Berikut adalah beberapa pendekatan yang disebutkan oleh mereka:

1. Pendekatan Berbasis Model (*Model-Driven Approaches*): Pendekatan ini berfokus pada pembuatan model dan pengembangan sistem yang direpresentasikan dalam dokumen. Tujuan dari pendekatan ini adalah memvalidasi sistem yang akan dikembangkan.
2. Analisis Terstruktur (*Structured Analysis*): Pendekatan ini menggunakan teknik *model-driven* yang fokus pada proses untuk mendefinisikan persyaratan bisnis dalam sebuah sistem.
3. Analisis Berorientasi Objek (*Object-Oriented Analysis*): Pendekatan ini menggunakan teknik yang berhubungan dengan data dan proses dalam pembentukan konstruksi yang disebut "objek". Pendekatan ini menggambarkan interaksi objek dalam sistem, termasuk perspektif, struktur, dan operasionalnya.
4. Rekayasa Informasi (*Information Engineering*): Pendekatan ini adalah pendekatan berpusat pada data dan responsif terhadap proses dalam menggambarkan data dan proses sistem.
5. Prototyping Penemuan (*Discovery Prototyping*): Pendekatan ini digunakan untuk mengidentifikasi persyaratan bisnis dari pengguna dengan membuat implementasi yang efektif. Pendekatan ini memungkinkan para pengguna bereaksi terhadap implementasi yang dihasilkan tanpa efek samping dan kesalahan terhadap persyaratan yang telah ditentukan.

2.1.3 Definisi Pendaftaran

Sesuai dengan Kamus Besar Bahasa Indonesia (KBBI), pendaftaran didefinisikan sama seperti tindakan menuliskan nama, alamat, serta sejenisnya pada suatu daftar. Sebagaimana dijelaskan oleh Departemen Pendidikan dan Kebudayaan (2021), pendaftaran merupakan suatu proses atau cara untuk membuat atau mendaftarkan sesuatu; penulisan nama, alamat, serta sejenisnya pada daftar. Mengacu pada saran para ahli di atas, dapat ditarik kesimpulan bahwa pendaftaran adalah langkah mencatat informasi identitas pendaftar ke dalam suatu media penyimpanan yang digunakan dalam proses pendaftaran.

Dari pernyataan yang disampaikan sebelumnya, dapat disimpulkan oleh penulis bahwa pendaftaran merupakan suatu proses di mana informasi pribadi pendaftar direkam ke dalam suatu media penyimpanan yang digunakan dalam proses pendaftaran.

2.1.3.1 Pengertian Pendaftaran Produk

Pendaftaran produk merupakan suatu proses mendaftarkan produk baru ke dalam suatu wilayah geografis atau yurisdiksi tertentu, biasanya melalui lembaga pemerintah sebagai pihak yang melindungi hak kekayaan intelektual, seperti merek dagang atau paten. Pendaftaran produk harus sesuai dengan prosedur yang berlaku di perusahaan atau organisasi. Proses pendaftaran produk harus dilakukan dengan benar sehingga dapat mengurangi resiko kesalahan didalam proses penginputan produk.

2.1.3.2 Tujuan Pendaftaran Produk

Tujuan pendaftaran produk adalah memberikan hak eksklusif kepada pemilik produk, melindungi karya atau inovasi, mencegah penggunaan yang tidak sah, dan memberikan perlindungan hukum yang diperlukan. Selain itu, pendaftaran produk juga memiliki manfaat dalam hal perlindungan hukum,

identifikasi dan diferensiasi produk, peningkatan nilai aset bisnis, ekspansi dan perlindungan pasar, serta perdagangan internasional.

2.1.4 Definisi Produk

2.1.4.1 Pengertian Produk

Produk adalah suatu entitas yang beragam, yang dapat mempunyai sifat serta dapat dirasakan atau tidak dirasakan, dan mencakup berbagai elemen seperti kemasan, harga, reputasi badan usaha, dan layanan diberikan oleh perusahaan demi memenuhi harapan dan kebutuhan pembeli. Produk dapat dikelompokkan menjadi dua kategori, yaitu jasa dan produk. Produk jasa bersifat abstrak dan tidak berwujud, sedangkan produk barang bersifat konkret dan berwujud.

Dalam konsep Philip Kotler, produk didefinisikan sebagai hal yang dapat dipresentasikan kepada lingkungan bisnis agar menarik perhatian, dipunyai, digunakan, atau dipakai oleh konsumen, dengan tujuan untuk memenuhi keinginan atau kebutuhan yang ada.

Pembagian produk dapat dibagi menjadi beberapa kategori, antara lain:

a) Berdasarkan pada ketahanan dan keberwujudan

1) *Nondurable goods* (barang tidak tahan lama) merujuk pada barang-barang berwujud yang umumnya dikonsumsi dalam satu atau beberapa kali penggunaan, seperti sabun, pasta gigi, makanan, minuman ringan, dan sejenisnya.

2) *Durable goods* (barang tahan pakai) merujuk pada barang-barang yang dapat digunakan dalam jangka waktu yang panjang, seperti kulkas, mesin fotokopi, pakaian, dan lain sebagainya.

3) *Services* (layanan) merujuk pada produk abstrak, tidak terbagi, memiliki beragam jenis, serta dapat hilang atau tidak tetap, seperti layanan kecantikan, konsultasi hukum, dan pemeliharaan peralatan.

b) Produk Untuk Konsumen

1) Barang kebutuhan sehari-hari (*convenience goods*) merupakan barang yang sering dibeli oleh konsumen, seperti makanan, minuman, serta sejenisnya. Konsumen cenderung membeli barang-barang ini dengan cepat dan tanpa kesulitan.

2) Barang perbelanja (*shopping goods*) merupakan barang yang dibandingkan sebelum pembelian oleh konsumen berdasarkan karakteristik seperti kesesuaian, mutu, kost, dan mode. Contohnya termasuk baju, sepatu, produk kecantikan, dan lainnya. Konsumen biasanya melakukan perbandingan antara produk-produk ini sebelum memutuskan untuk membeli.

3) Barang khusus (*specialty goods*) adalah sejenis barang yang mempunyai ciri-ciri khusus dalam hal identifikasi mereknya. Jenis barang ini menarik perhatian sekelompok konsumen yang dengan sengaja melakukan upaya ekstra untuk memperolehnya. Diantaranya termasuk sepeda motor, kendaraan roda empat, handphone mewah, serta sejenisnya. Barang-barang ini biasanya dikaitkan dengan merek yang memiliki reputasi tinggi dan daya tarik khusus bagi sebagian konsumen yang memiliki preferensi dan keinginan yang spesifik.

4) Barang yang tidak diminati (*unsought goods*) merujuk pada barang-barang yang tidak familiar oleh konsumen yang tidak umum memikirkan untuk membelinya. Artinya, konsumen pasif dalam mencarinya atau tidak memiliki kebutuhan yang jelas untuk barang tersebut. Contohnya termasuk asuransi jiwa, peralatan pemadam kebakaran, dan produk-produk kesehatan yang khusus digunakan dalam kondisi darurat. Konsumen mungkin baru akan mempertimbangkan untuk membeli barang ini ketika situasi atau keadaan tertentu muncul, atau ketika mereka diperkenalkan dengan

keberadaan barang tersebut melalui pemasaran atau informasi yang relevan.

c) Produk Industry

1) Bahan dan komponen pengganti (*materials and parts*) mengacu pada barang-barang yang sepenuhnya terintegrasi ke dalam produk yang diproduksi oleh pembuat. Contohnya, produk-produk hasil pertanian seperti kapas, gandum, beras, dan sejenisnya. Barang-barang ini digunakan sebagai komponen atau bahan baku dalam langkah akhir produksi untuk menghasilkan produk akhir yang lebih kompleks. Dalam konteks ini, bahan dan suku cadang berperan penting dalam menciptakan produk yang dihasilkan oleh produsen.

2) Barang modal (*capital items*) merujuk pada barang-barang awet yang berperan dalam mendukung pengembangan atau pengelolaan produk jadi. Contohnya adalah bangunan-bangunan yang digunakan dalam pendirian pabrik seperti ruang kerja, lokasi produksi, mesin pabrik, serta sejenisnya. Barang-barang modal ini merupakan investasi jangka panjang yang digunakan untuk mendukung proses produksi dan operasional perusahaan. Mereka berperan penting dalam infrastruktur dan fasilitas yang diperlukan untuk menghasilkan produk akhir.

3) Jasa bisnis dan suplai (*supplies and business service*) merujuk pada produk dan layanan dalam jangka waktu pendek yang berperan dalam membantu dalam pengembangan atau pengelolaan produk akhir. Contohnya adalah persediaan, yaitu peralatan yang digunakan dalam kegiatan produksi. Sebagai contoh oli mesin, bahan bakar otomotif, peralatan tulis, serta sejenisnya. Barang dan jasa ini merupakan komponen penting yang diperlukan dalam menjalankan proses produksi dan operasional bisnis secara efisien.

2.1.4.2 Pengertian Kualitas Produk

Kualitas produk mencerminkan kapabilitas produk untuk menjalankan manfaat-manfaatnya secara efektif. Kapabilitas ini termasuk faktor-faktor seperti ketahanan, kredibilitas, ketepatan hasil yang diperoleh, kemudahan pengoperasian serta perbaikan, serta komponen lain yang memberikan nilai tambah dalam keseluruhan produk.

Kualitas produk menjadi prioritas utama bagi konsumen, yang mengharapkan standar yang tinggi dalam pengeluaran uang mereka. Mereka menghindari membeli secara sembarangan, karena khawatir bahwa biaya dan usaha yang mereka keluarkan tidak sepadan dengan kualitas produk yang akan mereka dapatkan.

A. Indikator Kualitas Produk

Untuk mengukur standar kualitas produk, perlu dilakukan survei terhadap konsumen. Aspek-aspek yang perlu diperhatikan meliputi:

a) Penyimpanan produk oleh produsen, terutama untuk produk makanan, harus memenuhi standar kebersihan dan keamanan.

b) Penampilan produk yang terkait dengan kualitas desain, yang juga menjadi identitas produk.

c) Keandalan produk, yakni seberapa besar kemungkinan produk mengalami kerusakan atau kegagalan.

d) Fungsi produk yang harus sesuai dengan kebutuhan konsumen.

e) Kelengkapan dan garansi produk sebagai fasilitas tambahan yang meningkatkan nilai produk.

f) Daya tahan produk, sehingga konsumen menginginkan produk yang awet dan tidak mudah rusak.

g) Kesesuaian produk dengan spesifikasi yang ditawarkan kepada pelanggan.

h) Kesesuaian estetika produk dengan preferensi konsumen.

i) Kualitas pelayanan yang mencakup garansi, pelayanan purna jual, dan servis.

j) Estetika produk yang berkaitan dengan nilai estetika yang subjektif dan terkait dengan preferensi individu konsumen.

B. Korelasi antara Kualitas Produk dan Keputusan Pembelian

Meningkatkan kualitas produk merupakan prioritas utama karena kualitas produk memainkan peran krusial dalam keputusan pembelian konsumen. Perusahaan dengan produk berkualitas terbaik akan mengalami pertumbuhan yang pesat dan pada akhirnya akan lebih berhasil daripada perusahaan lain.

2.1.5 Elisitasi

Hasil dari wawancara dan pengamatan akan dikumpulkan, diklasifikasikan, dan diproses melalui tiga tahapan Elisitasi. Menurut Sarah Riwanda Shofroh (2015) dalam artikel *i-learning* menjelaskan bahwa teknik elisitasi terbagi menjadi tiga tahap, yaitu:

1. Tahap Pertama; Tahapan awal dalam perencanaan pengembangan sistem yang diusulkan berdasarkan hasil pengamatan dan wawancara.
2. Tahap Kedua; Tahapan selanjutnya adalah mengklasifikasikan hasil dari tahap pertama untuk memisahkannya mengacu pada metode MDI. Metode MDI merujuk pada pendekatan pemisahan antara rencana pengembangan sistem terbaru dengan desain yang dapat dieksekusi oleh penulis. Penguraian lebih lanjut terkait Metode MDI:
   a. M dalam MDI atau *Mandatory* (Penting) berarti bahwa kebutuhan tersebut yang perlu ada dan tidak boleh dihapus saat pembuatan sistem terbaru.
   b. Dalam MDI atau *Desirable* (Diperlukan), kebutuhan tersebut opsional atau kurang penting. Namun, jika kebutuhan tersebut diimplementasikan dalam pembentukan sistem, akan meningkatkan kualitas dan kesempurnaan sistem.

c. Dalam MDI atau *Inessential* (Tidak Penting), kebutuhan tersebut tidak termasuk dalam ruang lingkup sistem yang sedang dibahas atau merupakan bagian yang berada di luar sistem tersebut.

3. Tahap ketiga melibatkan pengurangan kebutuhan sistem yang telah disesuaikan sebelumnya. Berikutnya, setiap kebutuhan Yang masih ada akan diurutkan kembali menggunakan metode TOE (*Tangible*, *Observable*, *and Estimable*) yang berarti kebutuhan harus dapat diukur secara nyata, dapat diamati, dan dapat diperkirakan.
   a. *Technical* (T) merupakan faktor yang terkait dengan metode atau teknik pembuatan kebutuhan sistem yang diusulkan.
   b. *Operational* (O) adalah aspek yang berkaitan dengan penggunaan kebutuhan dalam sistem yang akan dikembangkan, termasuk tata cara penggunaan sistem tersebut.
   c. *Economic* (E) adalah aspek yang berkaitan dengan kesimpulan mengenai pengeluaran yang dibutuhkan untuk membangun sistem tersebut.

   Dalam pendekatan TOE ini, terdapat pembagian berikut:

   a) *High* (H): Menunjukkan tingkat kesulitan yang tinggi dalam pembuatan dan penggunaannya, serta memiliki biaya yang tinggi. Oleh karena itu, kebutuhan tersebut dapat dieliminasi.

   b) *Middle* (M): Memiliki tingkat kekerasan yang moderat dan dapat dilakukan.

   c) *Low* (L): Mudah untuk dikerjakan.

2.1.6 Metode Penelitian

Metode penelitian merupakan suatu pendekatan atau strategi yang terencana dan digunakan untuk mengumpulkan data dan fakta yang diperlukan dalam penelitian. Pandangan Sugiono (2012), "Metode penelitian secara mendasar merupakan suatu pendekatan rasional dalam mengumpulkan

data dengan tujuan jelas serta manfaat tertentu." Secara umum, terdapat tiga jenis penelitian, yaitu:

1. Jenis pertama adalah penelitian bersifat penemuan, di mana data yang didapat benar-benar baru dan sebelumnya belum pernah ada atau diketahui. Maksud dari jenis penelitian ini bertujuan untuk mengungkap temuan-temuan baru atau mengungkapkan informasi yang belum terungkap.
2. Jenis kedua adalah penelitian bersifat pembuktian, di mana data yang sudah diperoleh dimanfaatkan untuk membuktikan kebenaran atau validitas suatu informasi. Maksud dari jenis penelitian ini yaitu untuk memberikan bukti yang kuat atau menguji hipotesis yang ada.

Menurut penjelasan yang disampaikan oleh Prof. DR. Lexy J. Moleong pada tahun 2018 , metode kualitatif merujuk pada suatu prosedur yang menghasilkan data deskriptif dalam bentuk kalimat tertulis atau lisan mengenai individu dan perilaku yang dapat diobservasi.

Sebagaimana dijelaskan oleh Ety Rochaety (2007), penelitian deskriptif adalah suatu tindakan penelitian yang bertujuan untuk mengetahui karakteristik nilai-nilai variabel secara terpisah, baik itu satu variabel (*independent*) maupun lebih, tanpa melakukan perbandingan atau menghubungkannya dengan variabel lain.

Menurut pernyataan McMillan & Schumacher (2012), pendekatan kualitatif adalah suatu metode penelitian yang melibatkan investigasi langsung karena peneliti umumnya mengumpulkan data dengan berinteraksi dan bertatap muka langsung dengan individu di lokasi penelitian.

2.1.7 Aplikasi *Mobile*

Aplikasi *mobile*, juga dikenal sebagai *Mobile Application* atau *mobile-apps*, merupakan perangkat lunak yang dapat beroperasi secara mandiri dan independen dari perangkat keras lainnya atau sebagai entitas yang berdiri

sendiri. Aplikasi ini dirancang untuk dapat digunakan pada perangkat *mobile* seperti *smartphone*.

Mengacu penjelasan yang dikemukakan oleh Vijay Kumar Velu (2016: 3), pada sebagian perkembangan teknologi yang menarik di masa mendatang dalam konteks aplikasi seluler adalah pengembangan aplikasi seluler yang berfokus pada platform *Android* Dalam situasi ini, aplikasi memiliki kemampuan untuk mendengarkan sinyal suara dari dunia fisik dan meresponsnya. Vijay (2016: 3) juga mengkategorikan aplikasi secara umum menjadi beberapa jenis:

1. Aplikasi *Native*

Aplikasi *native* didapatkan melalui *platform* aplikasi untuk sistem operasi seluler yang relevan. Aplikasi ini didesain khusus untuk *platform* tertentu dan dapat memaksimalkan seluruh fitur perangkat, seperti kamera, GPS, daftar kontak telepon, dan lain-lain. Sebagian keunggulan lainnya, seperti yang dinyatakan oleh Jeff McWherter dan Scott Gowell (2012: 22), adalah kapbalitas aplikasi native untuk mengakses konten tanpa koneksi internet. Meskipun data disimpan di perangkatl, aplikasi ini dapat memanfaatkan layanan latar belakang untuk memeriksa konektivitas internet yang tersedia pada perangkat dan meminta izin pengguna untuk mengunduh pembaruan data.

2. *Mobile Web Apps*

Mobile Web Apps merupakan jenis aplikasi yang tidak bersifat asli (non-asli). Pengguna diberikan izin akses yang sama seperti ketika mengakses halaman *web* lainnya, namun telah ditingkatkan secara spesifik untuk perangkat *mobile*. Aplikasi ini berjalan di dalam *browser web*, walaupun tidak tampak dalam jendela *browser*. Keunggulan dari jenis aplikasi ini terletak pada tingkat kompatibilitas yang tinggi, karena dapat diakses pada berbagai platform. Akan tetapi, guna mengaksesnya,

perangkat harus terkoneksi dengan internet, sehingga tidak memungkinkan untuk menyimpan data secara *offline*.

3. Hybird Apps

Aplikasi hibrida mencoba menggabungkan keunggulan dari kedua pendekatan sebelumnya. Aplikasi ini memanfaatkan kemampuan komputasi dari sisi server, selain dari itu menganggap perangkat sebagai antarmuka pengguna semata. Aplikasi hibrida memiliki komponen asli yang diinstal di perangkat dan dapat memanfaatkan fitur lokal seperti aplikasi asli. Keuntungan dari jenis aplikasi ini adalah proses pengembangannya lebih sederhana dan lebih hemat biaya. Namun, API yang terkait dengan fitur-fitur khusus perangkat memiliki keterbatasan dibandingkan dengan aplikasi asli.

Berdasarkan penjelasan di atas, dapat ditarik kesimpulan bahwa aplikasi seluler adalah jenis aplikasi yang dapat diakses pada perangkat bergerak dan terhubung ke internet. Dengan demikian, pengguna dapat menggunakan aplikasi seluler ini di berbagai lokasi dan kapan saja sesuai kebutuhan mereka.

2.1.8 *System Development Life Cycle* (SDLC)

System development life cycle (SDLC) merujuk pada serangkaian tahapan yang bertujuan untuk memperoleh wawasan tentang bagaimana sistem informasi dapat memenuhi kebutuhan bisnis. Tahapan ini mencakup rancang sistem, pembangunan, dan pengiriman sistem kepada pengguna, sesuai dengan penjelasan yang diberikan oleh Tegarden, Dennis, dan Wixon pada tahun 2013.

SDLC mengacu pada empat tahapan pokok yang melibatkan perencanaan , analisis , desain , dan pelaksanaan . Semua tahapan tersebut melibatkan serangkaian langkah yang mengadopsi metode tertentu untuk

mencapai pencapaian sesuai dengan target yang ditetapkan. Penjelasan singkat berikut ini memberikan gambaran tentang keempat tahap tersebut.

1. Perencanaan (*Planning*)

   Tahap perencanaan merupakan elemen kunci dalam memahami alasan di balik pembangunan sistem informasi dan menentukan pendekatan yang akan digunakan oleh tim proyek dalam membangun proyek tersebut.

2. Analisis (*Analysis*)

   Tahap analisis memberikan tanggapan terhadap pertanyaan mengenai pengguna, fungsi, dan konteks penggunaan sistem. Pada tahap ini, tim proyek juga melakukan penyelidikan terhadap sistem yang sudah ada, mencari peluang untuk perbaikan, dan mengembangkan gagasan baru untuk sistem yang akan dikembangkan.

3. Perancangan (*Design)*

   Pada tahap perancangan melibatkan penentuan bagaimana sistem akan beroperasi, termasuk aspek-aspek seperti *hardware*, *software*, sistem jaringan (antarmuka pengguna), formulir beserta laporan (database dan file yang diperlukan oleh aplikasi).

4. Implementasi (*Implementation*)

   Pada tahapan yang terakhir dalam SDLC merupakan tahap implementasi, di mana sistem telah selesai dibangun. Implementasi menjadi perhatian utama karena secara keseluruhan, tahap ini menguras waktu dan biaya yang paling banyak karena mencakup pengujian dan penyiaran seluruh sistem.

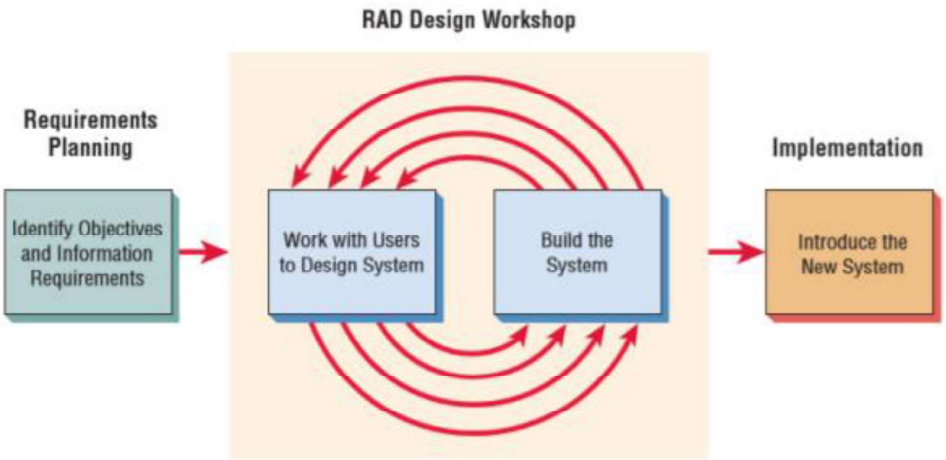
2.1.9 Metodologi Pengembangan Sistem

Penerapan . *System Development Life Cycle* (SDLC) dapat digunakan sebagai metodologi pengembangan sistem yang terstruktur. Ada berbagai metode untuk mengelompokan metodologi tersebut, yang ditentukan oleh fokus pada proses bisnis atau data yang relevan untuk bisnis tersebut. Akan

tetapi, dalam penelitian ini, digunakan pendekatan berorientasi pada kecepatan pengembangan dan penyampaian sistem yang dapat diakses oleh pengguna.

2.1.9.1 Rapid ApplicationDevelopment (RAD)

Dalam pengembangan aplikasi, salah satu pendekatan yang dipilih adalah *Rapid Application Development* (RAD), yang merupakan bagian dari *System Development Life Cycle* (SDLC). RAD adalah pendekatan pengembangan perangkat lunak yang memfokuskan pada pengembangan dalam waktu singkat dan mengadopsi pendekatan iteratif. Pada tahap awal pengembangan, model kerja dibentuk untuk mengidentifikasi kebutuhan pengguna, kemudian model tersebut ditingkatkan secara berulang dalam tahap pengembangan selanjutnya (Britton, 2000). Beberapa tahapan pengembangan RAD mencakup perencanaan kebutuhan, workshop desain RAD, dan implementasi (Britton, 2000). Gambar 2.1 menunjukkan tahapan-tahapan tersebut.



Gambar 2.1 Tahapan pendekatan RAD

(Kendall,2002)

Secara umum, pengembangan aplikasi biasanya memerlukan waktu paling sedikit 180 hari. Namun, dengan mamanfaatkan pendekatan ini, pengembangan dapat selesai di rentang waktu 30-90 hari. Disamping itu efisiensi waktunya, metode ini juga mampu menghasilkan produk yang berkualitas. Selain itu, metode ini dapat menetapkan pembatasan pada sistem untuk mencegah perombakan yang tidak diinginkan. Metode ini juga memungkinkan agar memperoleh rancangan yang disetujui oleh pengguna serta mudah dikembangkan di masa depan (Noertjahyana, 2002).

Metode Pengembangan Aplikasi Cepat (*Rapid Application Development*/RAD) berusaha untuk menghadapi tantangan dari kelemahan-kelemahan metode desain terstruktur dengan mengadaptasi tahap-tahap dalam SDLC untuk secara cepat mentuntaskan elemen sistem yang sedang dalam proses pengembangan dan dikembalikan kepada pengguna. Melalui pendekatan ini, pengguna dapat memiliki pemahaman yang terbaik terhadap sistem dan dapat memberikan rekomendasi perbaikan pada aplikasi yang mengalami masalah, untuk memastikan sistem mendekati tujuan yang diinginkan. Ada tiga pendekatan pokok yang diterapkan dalam RAD:

1. Pengembangan bertahap (*Phased Development)*

   Penggunaan metode pengembangan secara bertahap (phased development) dapat memisahkan sistem ke dalam beberapa bagian yang akan dikembangkan langkah demi langkah. Ketika tahap analisa, melakukan pengenalan terhadap konsep sistem secara menyeluruh, mengikutsertakan anggota tim proyek, user, dan infrastruktur pendukung, sambil mengkategorikan kebutuhan menjadi beberapa bagian yang berbeda. Kebutuhan yang terpenting dan dasar akan terdapat dalam komponen awal dari sistem. Ilustrasi mengenai metode pengembangan berfase dapat dilihat pada Gambar 2.2.

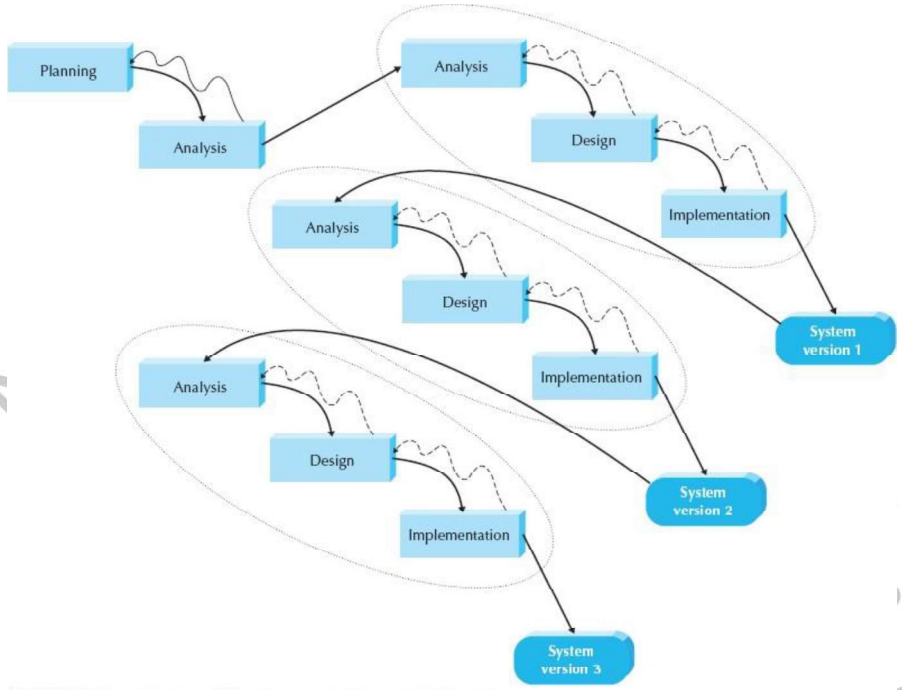


FIGURE 1-4 A Phased Development–based Methodology

Gambar 2.2 *Phased Development*

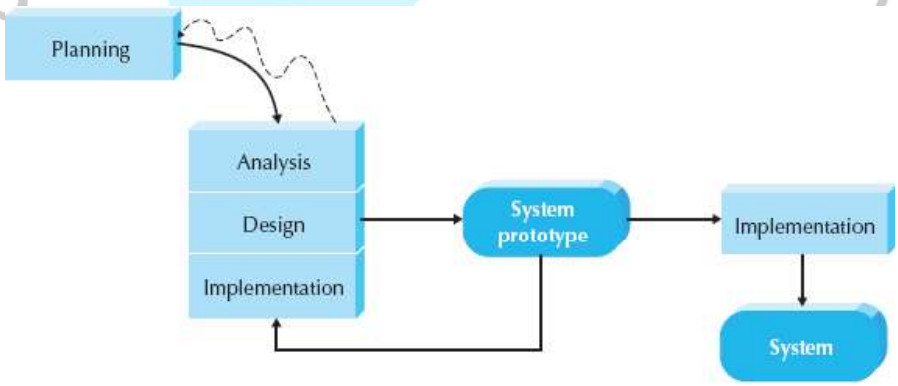
Pada saat tahap implementasi dilakukan pada bagian awal, langkah selanjutnya yaitu melanjutkan ke bagian berikutnya. Melakukan analisa lanjutan mengacu pada identifikasi kebutuhan sebelumnya, menginkorporasikannya dengan ide-ide baru, dan memperhatikan tantangan yang timbul dari pengalaman pengguna pada iterasi sebelumnya. Sesudah menyelesaikan pengembangan bagian saat ini, lanjutkan ke bagian berikutnya hingga semua bagian selesai.

Keunggulan dari penggunaan metode pengembangan berfase ini adalah kemampuannya dalam memperoleh sistem yang dapat dipakai oleh pengguna dengan cepat. Dengan metode ini memungkinkan pelaksanaan proses bisnis yang dibutuhkan oleh pengguna dapat dimulai lebih awal sebelum aplikasi sepenuhnya

selesai, meskipun sistem belum mampu sepenuhnya menjalankan semua fiturnya.

2. Pembuatan prototipe (*Prototyping*)

Pada penggunaan pembuatan metode prototipe mengikutsertakan penerapan analisia, desain, dan implementasi secara simultan, dan ketiga penerapan tersebut diulang beberapa saat hingga sistem diperbaiki secara keseluruhan. Dalam metode yang digunakan, analisa dan desain awal dari sistem dikembangkan dan langsung diimplementasikan dalam sistem uji coba yang berisi program yang masih belum sepenuhnya selesai. Biasanya, bpada agian sebelumnya yang sudah berjalan dimasukkan ke dalam sistem percobaan awal, yang kemudian diperlihatkan kepada pengguna atau perusahaan untuk mendapatkan umpan balik yang akan digunakan dalam proses analisis ulang, perancangan ulang, dan implementasi ulang pada percobaan berikutnya. Proses ini berlangsung secara berkelanjutan hingga sistem memenuhi persyaratan fungsionalitas dari pengguna atau perusahaan. Setelah pengujian sistem mencapai tingkat kesempurnaan, sistem tersebut diimplementasikan pada sistem yang sebenarnya. Rincian mengenai metode prototyping bisa terlihat pada Gambar 2.3.

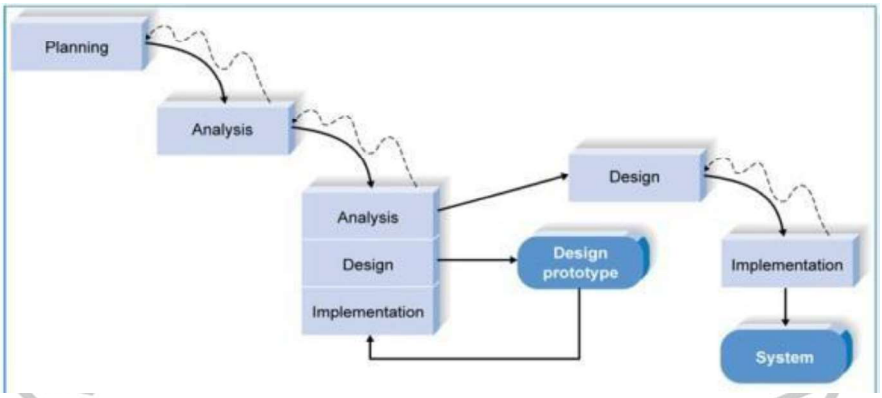


Gambar 2.3 *Metode Prototyping*

Manfaatnya menggunakan metode ini adalah bahwa sistem bisa segera dipakai oleh pengguna, walaupun belum dapat dijalankan secara menyeluruh di perusahaan yang terlibat. Pengujian sistem tersebut dapat memberikan kepercayaan kepada pengguna jika tim proyek telah melakukan pekerjaan sejak awal dan tidak membutuhkan waktu lama untuk implementasinya. Selain itu, dapat juga membantu memenuhi kebutuhan perusahaan secara konkret. Tidak seperti tim proyek yang harus mengerti semua aspek spesifikasi kebutuhan perusahaan, metode ini mempunyai keuntungan karena pengguna dapat langsung berinteraksi dengan sistem uji coba dan memahami kemampuan serta keterbatasannya.

3. *Throwaway Prototyping*

Dalam pengembangan melalui uji coba sistem, metode *throwaway prototyping* serupa dengan metode *prototyping*. Dengan metode ini, tahap pengkajian tetap dilakukan untuk memperoleh informasi dan mengembangkan gagasan konsep sistem yang akan dikembangkan.. Namun, sesudah melakukan pengujian sistem, besar kemungkinan pengguna yang mencobanya masih belum sepenuhnya memahami fitur yang mereka harapkan. Sistem uji coba yang dibuat bukanlah produk yang siap digunakan secara langsung oleh pengguna, melainkan masih memerlukan perbaikan dan penyesuaian. Sistem penguji coba ini juga hanya menyediakan sebagian kecil informasi yang bisa dipahami oleh pengguna. Lebih lanjut terlihat pada Gambar 2.4 dari metode *throwaway prototyping*.



Gambar 2.4 *Metode Throwaway Prototyping*

Sebuah sistem yang dibangun dengan metode ini mungkin bergantung pada berbagai konfigurasi uji coba yang berbeda, baik dalam fase analisis maupun fase desain. Tiap uji coba yang digunakan bertujuan untuk mengurangi risiko dalam sistem dengan memverifikasi penyelesaian masalah utama sebelum sistem yang sebenarnya diimplementasikan. Setelah semua hambatan dalam sistem telah diatasi, proyek akan melanjutkan ke tahap desain dan implementasi. Hal ini yang menjadi pembeda antara metode ini dengan metode *prototyping*.

2.1.10 UML (*Unified Modeling Language*)

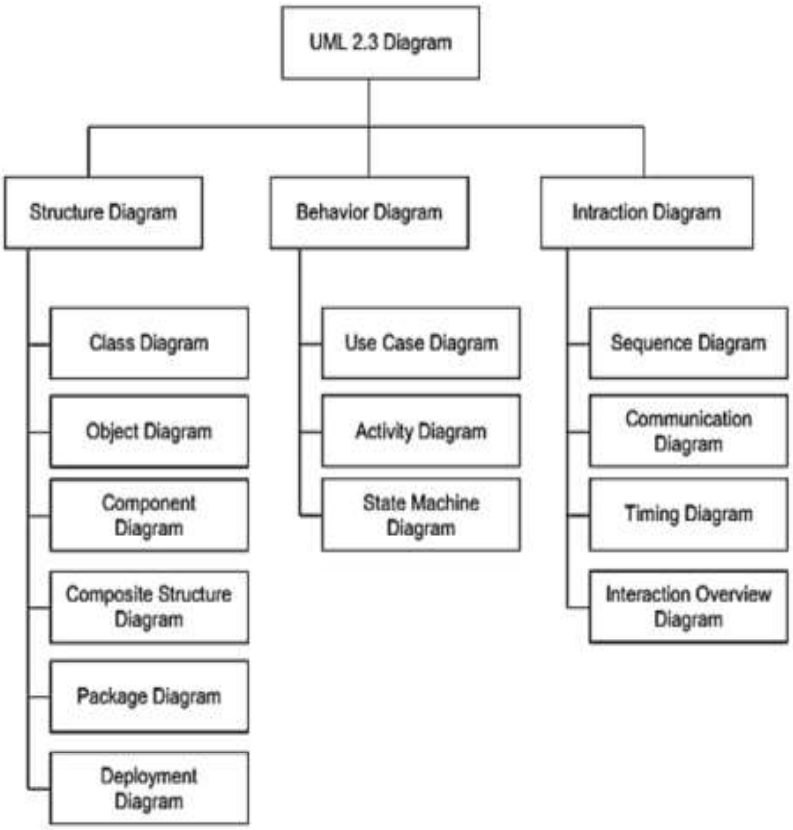
Dalam kutipan dari Rosa A. S & M. Shalahudin (2013: 26), dijelaskan bahwa *Unified Modeling Language* (UML) merupakan sebuah pola bahasa pemodelan yang digunakan dalam pengembangan perangkat lunak dengan pendekatan berdasarkan pada objek didalam pemrograman. UML menggunakan kombinasi antara teks dan diagram sebagai sarana untuk menggambarkan dan berkomunikasi mengenai perangkat lunak yang sedang dalam proses pengembangan.

Tujuan umum dari UML menurut Grady Booch, dkk (2004: 3), adalah untuk merancang spesifikasi, menciptakan visualisasi, membangun, dan

mengarsipkan rangkaian dasar dari sebuah sistem perangkat lunak. Menurut penjelasan Ian Sommerville (2015: 218), UML juga dapat digunakan untuk menciptakan dokumentasi berkaitan dengan keperluan perangkat keras dan perangkat lunak sebuah sistem, serta mengilustrasikan bagaimana elemen perangkat lunak disalurkan pada berbagai platform perangkat keras.

Berdasarkan beberapa opini yang telak dikemukakan, Dari uraian tersebut, dapat diambil kesimpulan bahwa UML adalah sebuah bahasa standar yang digunakan untuk menggambarkan struktur dan interaksi suatu sistem dengan menggunakan komponen-komponen yang terlibat, dengan tujuan menunjukkan kaitannya antara perangkat lunak dan perangkat keras.

Dalam konteks ini, pada tahap ini, analisis kebutuhan yang telah terkumpul pada tahap sebelumnya dijadikan sebagai dasar untuk mengembangkan model UML. Setelah proses penyusunan UML selesai, tahap selanjutnya dapat dilanjutkan. Dalam publikasinya, menurut Rosa A. S & M. Shalahudin (2011: 117) UML diklasifikasikan ke dalam tiga kelompok yang terdiri dari 13 jenis diagram.



Gambar 2.5 *Diagram* UML(Rosa, 2011: 121)

Structure diagrams mencakup sekelompok diagram yang dimanfaatkan untuk mengilustrasikan struktur yang tidak berubah-ubah dari sebuah aplikasi. Behavior diagrams merupakan sekelompok diagram yang dipakai untuk mendeskripsikan serangkaian perombakan atau perilaku dari suatu aplikasi. Interaction diagrams merupakan kumpulan diagram yang dimanfaatkan untuk menjelaskan interaksi dan komunikasi antara aplikasi ke aplikasi lainnya.

Didalam buku Reza B'Far pada tahun 2004 (halaman 50), UML adalah sebuah alat yang banyak digunakan yang dapat memiliki penafsiran yang bervariasi tergantung pada cara implementasinya dalam suatu *platform* spesifik. Ian Sommerville didalam bukunya tahun 2015 (halaman 205)

menceritakan bahwa dibutuhkan tingkat rincian dalam model desain akan bervariasi tergantung pada teknik desain yang sedang digunakan.

Berikut ini penjelasan dari masing-masing diagram:

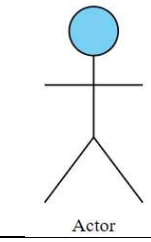
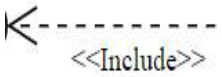
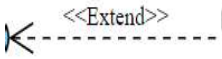
1. *Use case Diagram*

Use case merupakan sebuah alat yang sangat bermanfaat dalam memaparkan bagaimana suatu sistem digunakan dan persyaratan yang harus dipenuhi menurut penjelasan dalam bukunya Reza B'Far dalam bukunya tahun 2004 (halaman 165). Kemudian, menurut Rosa A.S & Shalahuddin dalam bukunya tahun 2011 (halaman 130), penggunaan *use case* dengan tujuan untuk mengenali fungsi-fungsi yang terdapat dalam sistem informasi dan mengenali pengguna yang bersyarat. Penjelasan Janis Osis dan Uldins Donnis dalam bukunya tahun 2017 (halaman 23) bahwa diagram *use case* dimanfaatkan untuk mempersiapkan dan membangun komponen yang dinamis dari suatu sistem dengan memperlihatkan rentetan *use case*, aktor, dan hubungan di antara mereka.

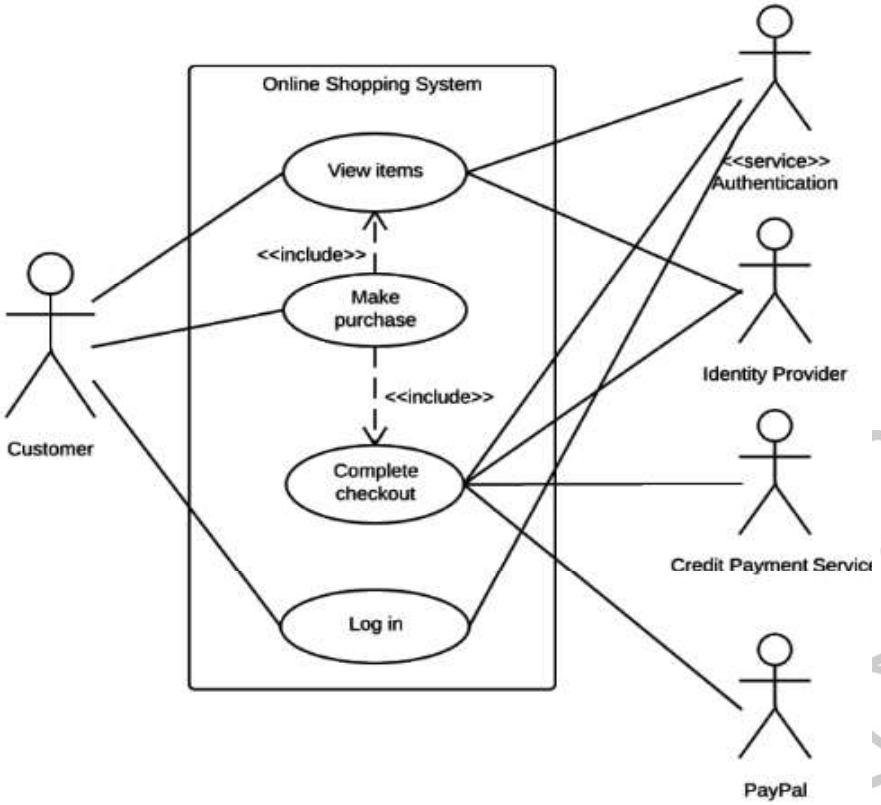
Menurut Alistair Cockburn, seperti yang dikutip oleh Pressman dalam bukunya tahun 2010 (halaman 155*), use case* dapat dijelaskan sebagai sebuah kesepakatan yang menggambarkan tingkah laku sistem. Kesepakatan ini mengacu pada bagaimana aktor menggunakan sistem berbasis komputer untuk mencapai berbagai tujuan. Dengan secara jelas menunjukkan betapa pentingnya *use case* dalam memudahkan pengguna dalam memahami dan pihak pengembang, dengan memvisualisasikan komunikasi antara pengguna dan sistem.

Penjelasan berikut ini adalah tanda dan simbol yang digunakan dalam Diagram *Use Case* yang tercantum dalam Tabel 2.1.

Tabel 2.1 Notasi dan Simbol *Use Case Diagram*

Simbol	Nama Simbol	Keterangan
	*Actor*	Merupakan simbol *use case* yang mewakili peran individu atau orang, entitas lain, atau perangkat yang terlibat dalam dialog dengan *use case.*
	*Use Case*	Representasi dan dialog antara sistem dan aktor
	Association	Representasi dan dialog antara aktor dan *use case*
	*Generalization*	Menunjukkan peran khusus aktor dalam *use case*.
	*Include*	Mengindikasikan satu *use case* sepenuhnya tergantung pada *use case* lainnya.
	*Extend*	Menggambarkan suatu *use case* dapat digunakan sebagai opsi tambahan dari *use case* lainnya berdasarkan pemenuhan kondisi tertentu.

Berikut adalah contohnya use case diagram Sistem Pendaftaran produk tercantum pada Gambar 2.6:



Gambar 2.6 Contoh *Use Case Diagram*

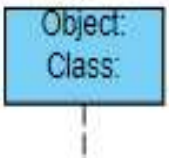
(https://www.lucidchart.com/pages/uml-use-case-diagram)

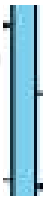
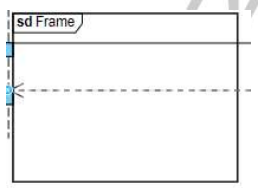
2. ***Sequence Diagram***

Pada penggunaan *sequence diagram* adalah untuk memvisualisasikan interaksi objek dalam suatu *use case* dengan menyajikan pesan-pesan yang dikirim dan pertukaran pesan antara objek serta linkup hidup objek tersebut (Rosa A.S & Shalahuddin, 2011: 137). Dikarenakan sebab itu, saat pembuatan sequence diagram, penting untuk mempelajari dengan seksama objek-objek yang terlibat dan metode-metode yang dipunyai oleh kelas yang ketika objek tersebut diinisialisasi menjadi objek tersebut. *Sequence diagram* mengilustrasikan rangkaian hubungan yang terjalin selama suatu *use case* tersebut (Ian Sommerville, 2015: 146).

Sesudah mengenali kejadian melalui analisis *use case* tersebut, seorang pemodel akan membangun *sequence diagram* yang merupakan pemetaan visual mengenai bagaimana peristiwa-peristiwa tersebut mengarahkan aliran informasi antar objek dalam urutan waktu (Roger S. Pressman, 2010: 197-198). Dengan kata lain, *sequence diagram* merupakan gambaran singkat dari *use case* yang menggambarkan kelas-kelas utama dan kejadian-kejadian yang mempengaruhi aliran perilaku dari satu kelas ke kelas lainnya. Dari penjelasan tersebut, dapat disimpulkan bahwa *sequence diagram* memiliki peran penting dalam menggambarkan aliran kejadian dari satu objek ke objek lainnya, karenanya memberikan pemahaman mengenai susunan objek tersebut.

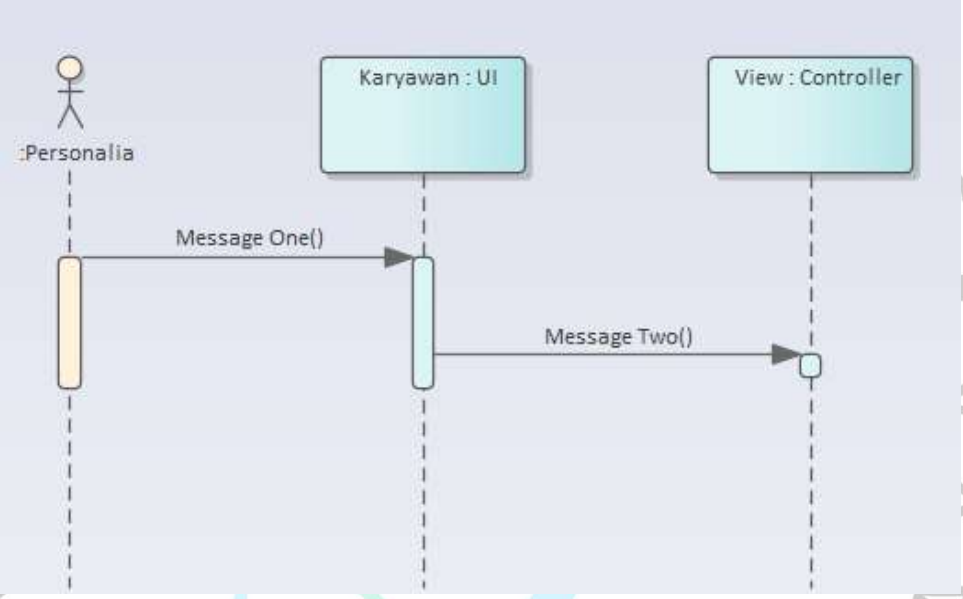
Tabel 2.2 Notasi dan Simbol *Sequence Diagram*

Simbol	Nama Simbol	Keterangan
	*Actor*	Merupakan pihak yang mendapat manfaat dari sistem namun berada di luar sistem. Actor terlibat dalam pertukaran pesan secara berurutan. Aktor dapat ditampilkan sebagai figur tongkat atau untuk non-manusia berbentuk persegi panjang dengan label "<<actor>>".
	*Object*	Suatu representasi *visual* dari objek yang terlibat dalam interaksi. *Object* biasanya digambarkan sebagai kotak dengan nama *Object* di dalamnya. *Object* berperan dalam urutan melalui

		pertukaran pesan, dan mereka berada di bagian atas diagram.
	*Lifeline*	Suatu representasi visual yang menunjukkan periode waktu keberadaan objek selama urutan. Pada titik di mana objek tidak lagi berinteraksi, *lifeline* ditandai dengan simbol X.
	*Execution Occurence*	Suatu representasi *visual* yang menunjukkan interval waktu di mana objek sedang melakukan aktivitas. Notasi ini ditempatkan di atas *lifeline* dan diberi label untuk menjelaskan jenis aktivitas yang dilakukan.
	*Message & Reply Message*	Untuk mengirim pengiriman informasi dari objek ke objek. *Message* ditandai dengan label pesan yang dikirim dan ditampilkan dalam garis penuh, sedangkan *Reply Message* diidentifikasi oleh nilai yang dikembalikan dan ditampilkan dengan garis terputus-putus.
	*Frame*	*Frame* digunakan untuk menggambarkan interaksi yang kompleks antara beberapa objek dalam *Sequence Diagram*. Biasanya, digambarkan sebagai sebuah kotak dengan label yang menjelaskan

		bagaimana interaksi tersebut terjadi. Beberapa contoh seperti penggunaan label sebuah frame antara lain loop, *opt, alt, par, break, seq,* dan *strict*.

Berikut contoh pemakaian *Sequence Diagram* seperti pada Gambar 2.7:



Gambar 2.7 Contoh *Sequence Diagram* (Dokumentasi Penulis)

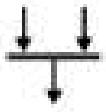
3. ***Activity Diagram***

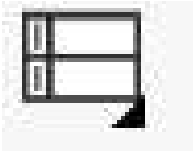
Penggunaan *Activity diagram* yaitu untuk menggambarkan visualisasi dinamis dari suatu sistem, yang menampilkan urutan atau percabangan tindakan, arus kontrol, dan nilai-nilai yang dihasilkan atau digunakan oleh tindakan-tindakan tersebut. Hal ini dijelaskan oleh Janis Osis dan Uldis Donnis (2017: 24). Dengan kata lain, activity diagram memberikan gambaran mengenai langkah-langkah aktivitas yang terjadi dalam pemrosesan internal suatu objek selama proses untuk mencapai perilaku yang diinginkan. Konsep ini dapat disarikan seperti yang dijelaskan oleh Reza B'Far (2004: 213).

Activity diagram memvisualisasikan jalur kerja atau aktivitas dalam sebuah aplikasi, sesuai dengan penjelasan dari Rosa A.S & Shalahuddin (2011: 134). Dalam bahasa yang lebih mudah dipahami, diagram aktivitas secara sederhana mengilustrasikan aktivitas yang terjadi dalam aplikasi, bukan aktivitas yang dilakukan oleh aktor. Menyajikan pemaparan grafis dari interaksi yang dilakukan dalam skenario tertentu, sebagai pelengkap informasi yang terdapat dalam *use case*. Berdasarkan hal tersebut, dapat disimpulkan bahwa activity diagram memiliki peran kunci dalam mempermudah dalam memahami bagaimana objek menjalankan proses internalnya, termasuk aktivitas, tindakan, dan aksi yang dilakukan.

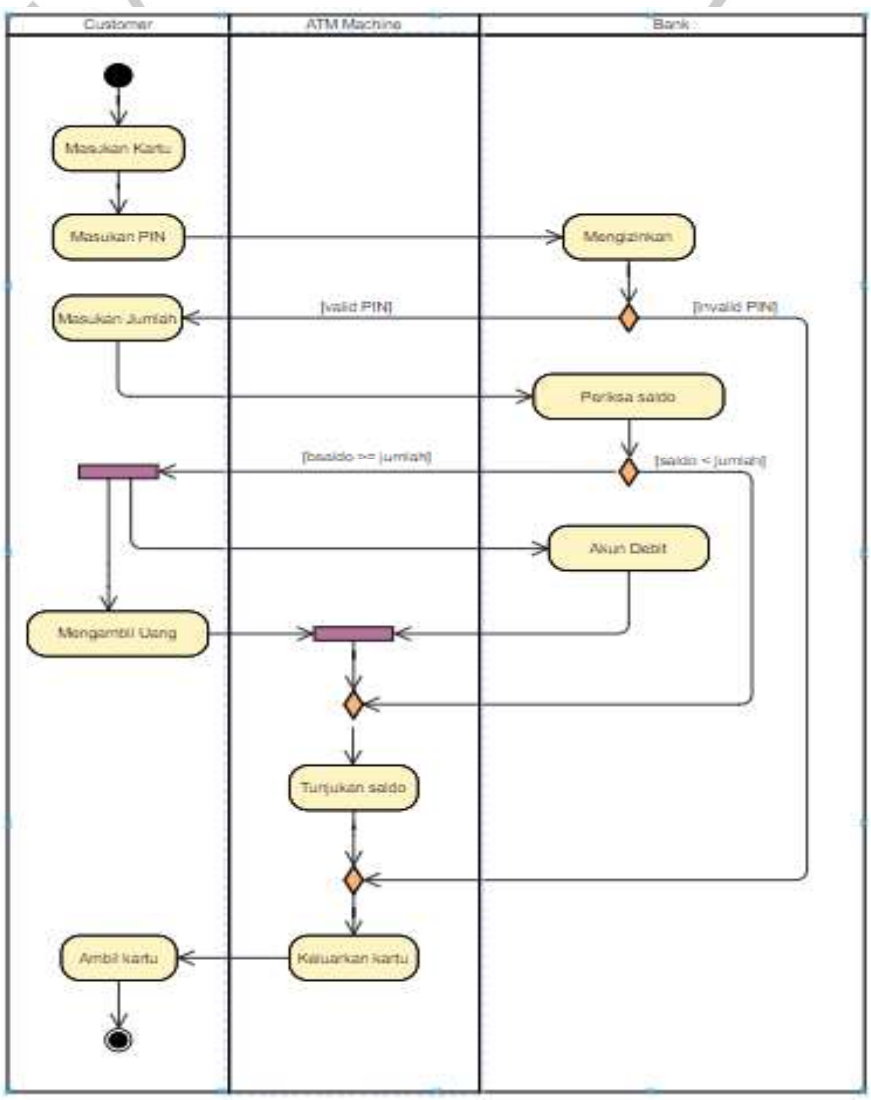
Tabel 2.3 Notasi dan Simbol *Activity Diagram*

Simbol	Nama Simbol	Keterangan
	*Initial Node*	Simbol visual yang menunjukkan titik awal dari rangkaian tindakan atau aktivitas.
	*Final-activity Node*	Simbol grafis yang menunjukkan akhir dari semua aliran kontrol dalam suatu aktivitas.
	*Activity*	Tanda ini digunakan untuk mengilustrasikan aktivitas yang dilakukan dalam sistem atau proses bisnis. Umumnya, aktivitas direpresentasikan oleh persegi panjang dan diberi label untuk menjelaskan jenis aktivitas yang dilakukan.

	*Control Flow*	Menunjukkan urutan eksekusi dari setiap aktivitas dalam suatu *Activity Diagram* dan menggambarkan bagaimana aktivitas yang saling terkait dan berurutan.
	*Decision Node*	Digunakan untuk mewakili kondisi pengujian dalam suatu diagram aktivitas. Fungsinya adalah memastikan bahwa aliran kontrol hanya melewati satu jalur tergantung pada hasil dari pengujian tersebut.
	*Fork Node*	Digunakan untuk membagi perilaku menjadi beberapa aliran aktivitas yang dilakukan secara bersamaan. Dengan menggunakan *fork node*, aktivitas dapat dijalankan secara paralel, memungkinkan beberapa aliran aktivitas untuk berlangsung secara simultan.
	*Join Node*	Digunakan untuk menggabungkan kembali serangkaian aktivitas yang berjalan paralel. Dengan menggunakan *join node*, aliran aktivitas yang sebelumnya terbagi menjadi beberapa aliran dapat digabungkan kembali menjadi satu aliran tunggal.

	*Swimlane*	Membagi diagram aktivitas menjadi baris dan kolom. Setiap baris mewakili entitas yang terlibat, dan setiap kolom mewakili aktivitas yang harus dilakukan.

Berikut contoh penggunaan Activity Diagram seperti pada Gambar 2.8



Gambar 2.8 Contoh *Sequence Diagram*

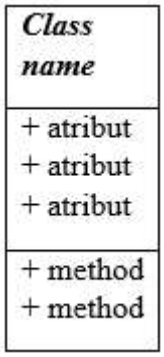
(Dokumentasi Penulis)

4. ***Class Diagram***

Selanjutnya, pada penggunaan *Class diagram* yaitu untuk pengertian kelas, mengilustrasikan kelas, dan interaksi antar kelas-kelas tersebut. Dalam diagram kelas, hubungan-hubungan digunakan untuk memvisualisasikan interaksi, kolaborasi, dan pewarisan antara kelas-kelas. Di samping itu, Bhuvan Unhelkar (2018: 21) menjelaskan pada kelas-kelas tersebut juga merupakan perwakilan tabel relasional, antarmuka pengguna, dan pengendali. Menurut Bhuvan Unhelkar (2005: 69), class diagram juga bermanfaat dalam pembuatan model basis data relasional yang efektif. Hal ini dikarenakan class diagram dapat dihubungkan dengan baik ke dalam format tabel relasional.

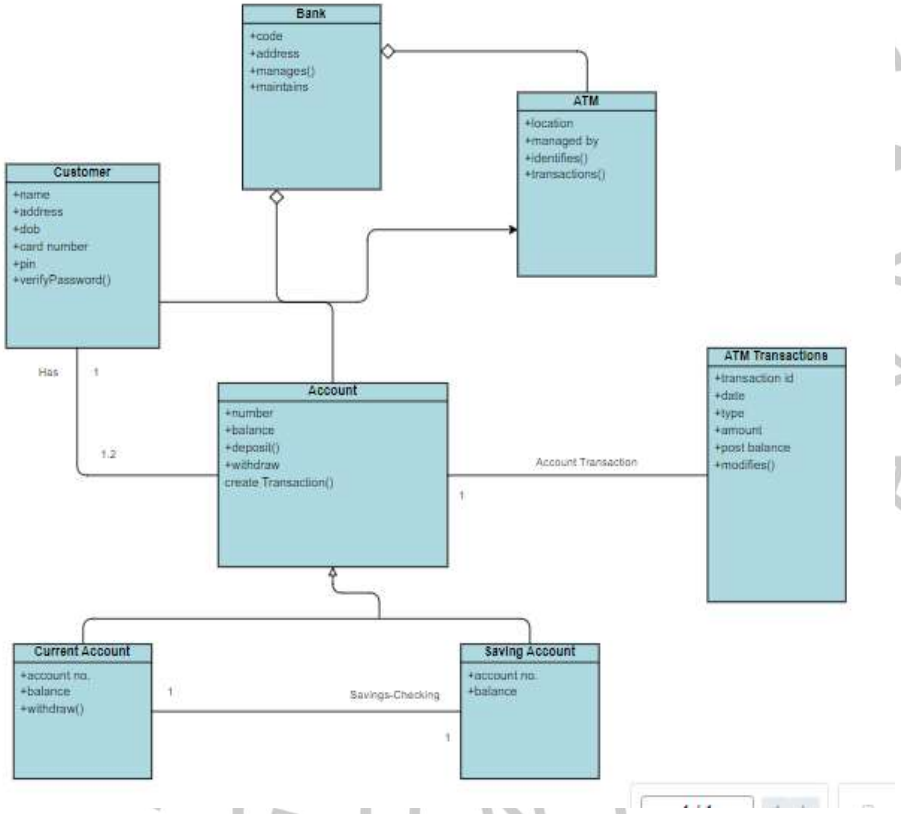
Lanjutan dari penjelasan Bernhard Rumpe (2017: 34), untuk memvisualisasikan struktur atau arsitektur suatu sistem yaitu menggunakan *class diagram*. Memiliki konsep kelas kegunaan yang luas dalam pemograman dan pemodelan, oleh sebab itu class diagram menjadi rangkaian inti yang membantu dalam menemukan kesalahan dalam berbagai kegiatan dan melacak persyaratan proyek. Reza B'Far (2004: 172) ia pun beropini jika class diagram merupakan salah satu diagram inti dalam pemodelan UML, karena menjadi basis untuk diagram penempatan dan komponen, serta memodelkan gambar kestrukturan suatu sistem. Dengan berdasarkan beberapa pandangan tersebut, dapat disimpulkan bahwa *class diagram* memiliki peran krusial dalam mengartikan kelas-kelas dan hubungan di dalam pengembangan sistem perangkat lunak.

Tabel 2.4 Notasi dan Simbol *Class Diagram*

Simbol	Nama Simbol	Keterangan
	*Class*	*Class* divisualisasikan dalam bentuk kotak yang terdiri dari tiga bagian. Pada bagian atas mengandung nama class, bagian tengah mendefinisikan atribut-atribut *class*, dan bagian akhir menjelaskan metode-metode yang dimiliki oleh *class* tersebu.
	Association	Asosiasi dalam *class diagram* direpresentasikan melalui suatu tautan yang menghubungkan antara blok-blok *class*.
	*Agregasi*	Relasi yang menunjukkan pada bagian ("terdiri atas...") atau sering dinamakan sebagai hubungan yang memiliki sebuah.
	*Composition*	Menunjukan hubungan di mana sebuah kelas bergantung pada kelas lain. Dengan kata lain, kelas tersebut memiliki relasi *composition*.
	Dependency	Menunjukkan bahwa sebuah kelas menggunakan atau bergantung pada kelas lain dalam operasinya.
	*Generalisasi* /Pewarisan	Suatu Hubungan hierarkis antara kelas-kelas. Pewarisan sebuah kelas dapat dari kelas lain dan mewarisi

		seluruh atribut dan metode dari kelas sumbernya, serta memperluas fungsionalitas. Dalam konteks ini, kelas yang diturunkan disebut sebagai anak dari kelas yang diwarisinya.

Berikut contoh penggunaan *Class Diagram* seperti pada Gambar 2.9.



Gambar 2.9 Contoh *Class Diagram*

(https://www.visual-paradigm.com/)

2.2 Tinjauan Studi

Dalam upaya mencari referensi dan mengumpulkan informasi untuk penelitian ini, peneliti melakukan studi literatur sebelumnya yang telah

dilakukan oleh beberapa peneliti terdahulu. Pendekatan deduktif digunakan dalam mengkaji contoh-contoh yang telah dilakukan sebelumnya serta menarik kesimpulan dari pembahasan yang sudah dilakukan. Beberapa kajian yang akan dijadikan rujukan dalam penelitian ini meliputi hal-hal berikut:

2.2.1 *Rapid Application Development*

Pada penelitian sebelumnya yang dilakukan oleh Nurman Hidayat dan Kusuma Hati pada tahun 2021, yang berjudul "Penerapan Metode *Rapid Application Development* (RAD) dalam Rancang Bangun Sistem Informasi Rapor Online (SIRALINE)" [7] dan dipublikasikan dalam Jurnal Sistem Informasi STMIK Antar Bangsa, dijelaskan bahwa penerapan metode RAD merupakan suatu proses pengembangan perangkat lunak yang mengikuti pendekatan *System Development Life Cycle* (SDLC) secara *waterfall* atau *linear sequential*. Metode ini memiliki keunggulan dalam hal kecepatan, akurasi, dan efisiensi penghematan biaya.

2.2.2 *Mobile Hybrid Application*

Dalam tahapan pengembangan aplikasi *mobile*, peneliti memilih teknik pembuatan aplikasi *cross-platform* yang bersifat *hybrid*, yang dapat digunakan pada beberapa *platform*. Teknik ini telah dibahas dalam penelitian terkait sebelumnya yang ditulis oleh Eksen Hidiansah pada Journal Pengembangan IT (JPIT) tahun 2018 dengan judul "*Mobile Hybrid Application* Sebagai Solusi Dalam Pelaporan Bencana Menggunakan *Framework Cordova*" [1].

Adapun terdapat beberapa penelitian studi terdahulu lainnya terkait perkembangan aplikasi *mobile* yang relevan dengan penelitian ini diantaranya:

1. Penelitian yang dilakukan oleh Muhammad Irsan pada tahun 2015 dengan judul "Rancang Bangun Aplikasi *Mobile Notifikasi Berbasis Android* Untuk Mendukung Kinerja di Instansi Pemerintahan".

Tujuan dari penelitian ini digunakan untuk menciptakan sebuah aplikasi mobile yang mampu mengirimkan pemberitahuan informasi secara langsung, sehingga memungkinkan pengawasan, pengambilan tindakan, dan pengiriman pemberitahuan kepada staf yang ditargetkan. Dalam kesimpulan penelitian ini adalah *Aplikasi Mobile Notifikasi* Berbasis *Android* membangun sebuah sistem yang mampu memberikan pemberitahuan mengenai kegiatan, dan rapat di lingkungan perkantoran kepada pengguna atau pegawai melalui webserver yang dikirim ke aplikasi klien. Selanjunya, pengguna atau pegawai yang sedang mengerjakan laporan tentang batas waktu pengumpulan laporan yang akan datang, memastikan pengumpulan laporan tepat waktu, karena aplikasi ini mampu memberikan notifikasi otomatis.

2. Selanjutnya penelitian yang dilakukan oleh pamela Alfa Adelia Darmadji yang berjudul "Rancang Bangun Aplikasi *Mobile* Untuk Notifikasi Jadwal Kuliah Berbasis Android (Studi Kasus: STMIK ProVisi Semarang)". Tujuannya adalah untuk membangun sebuah aplikasi mobile yang mempermudah pengguna untuk memperoleh akses ke jadwal kegiatan kuliah di STMIK ProVisi. Penelitian ini menghasilkan aplikasi *mobile* yang mempermudah mahasiswa untuk menerima pemberitahuan mengenai jadwal kuliah mereka di STMIK ProVisi. Aplikasi ini mampu memberikan akses data secara otomatis dan dilengkapi dengan fitur pemberitahuan untuk mengingatkan mahasiswa tentang jadwal kuliah yang harus diikuti.
3. Berikutnya penelitian yang dilakuakn pada tahun 2017 olwh Zara Dwi Rahayu dan Krisnawati yang berjudul "Analisis Dan Perancangan Notifikasi Untuk Kegiatan Forum Asisten Dengan Menggunakan *Cloud* SMS *Gateway*". Fokus penelitian ini untuk menganalisa dan mendesain aplikasi yang mampu dipergunakan serta memberikan notifikasi pada kegiatan Forum Asisten dengan memanfaatkan *Cloud*

SMS *Gateway*. Sebuah sistem informasi notifikasi kegiatan yang menggunakan *Cloud* SMS *Gateway* untuk mengirimkan pemberiahuan kepada para anggota Forum Asisten merupakan hasil akhir dari penelitian ini.