BAB II

LANDASAN TEORI

2.1 Perancangan Sistem

Perancangan melibatkan penggambaran, perencanaan, dan pembuatan sketsa atau pengaturan elemen terpisah ke dalam satu kesatuan yang utuh. Sistem dapat direncanakan dalam bentuk bagan alir, yang merupakan alat bantu grafik untuk menunjukkan urutan prosesnya. Dalam perancangan sistem informasi, umumnya terdapat dua jenis pemodelan sistem yang sering digunakan, yakni pemodelan terstruktur dan pemodelan berorientasi objek, keduanya memiliki pentingnya masing-masing dalam praktik perancangan sistem informasi. Pemodelan terstruktur sering kita kenal dengan bagan alir seperti aliran sistem informasi (*Flowchart System*), Diagram Konteks dan Diagram Alir Data (DAD). Sementara untuk pemodelan berorientasi objek umum kita lihat menggunakan *Unified Modeling Language* (*UML*). *UML* Digunakan untuk memberikan spesifikasi, menggambarkan, membangun dan mendokumentasikan dari sistem perangkat lunak (Julianto & Setiawan, 2019).

2.2 *Knowledge Management System*

Knowledge Management System merupakan salah satu solusi yang tepat untuk menerapkan SOP kerja, pengolahan pengetahuan baik yang lama ataupun yang baru dalam bentuk pengalaman dan modul, kesulitan dalam berbagi ilmu antar guru dan tempat penyimpanan yang terbatas (Perdana et al., 2019).

Knowledge Management System adalah suatu mekanisme dalam menyimpan, memelihara, dan mengorganisasikan informasi bisnis serta

pekerjaan yang berhubungan dengan penciptaan berbagai informasi yang menjadi aset intelektual organisasi yang permanen (Ishari et al., 2020).

Berdasarkan kedua pernyataan diatas dapat disimpulkan bahwa *Knowledge Management System* adalah solusi penting untuk menerapkan Standar Operasional (SOP) pekerjaan dan mengelola pengetahuan dalam organisasi. *Knowledge Management System* membantu organisasi mengumpulkan, menyimpan, dan mengorganisir informasi yang berkaitan dengan pekerjaan dan penciptaan aset intelektual. Sistem ini membantu organisasi mengatasi tantangan dalam berbagi pengetahuan antar anggota tim dan masalah tempat penyimpanan yang terbatas untuk informasi penting. Dengan demikian, penerapan *Knowledge Management System* dapat membantu organisasi menjadi lebih efisien.

2.3 Metode *Naive Bayes*

Naïve Bayes atau multinomial *naïve bayes* merupakan metode yang digunakan untuk mengklasifikasikan sekumpulan dokumen. Algoritma ini memanfaatkan metode probabilitas dan statistik yang dikemukakan oleh ilmuwan Inggris Thomas Bayes. Metode NB menempuh dua tahap dalam proses klasifikasi teks, yaitu tahap pelatihan dan tahap pengujian (klasifikasi). Pada tahap pelatihan dilakukan proses analisis terhadap sampel dokumen berupa pemilihan vocabulary, yaitu kata yang mungkin muncul dalam koleksi dokumen sampel yang mungkin dapat menjadi representasi dokumen. Selanjutnya adalah penentuan probabilitas *priority* bagi tiap kategori berdasarkan sampel dokumen. Pada tahap klasifikasi ditentukan nilai kategori

dari suatu dokumen berdasarkan term yang muncul dalam dokumen yang diklasifikasi (Wibisono et al., 2020).

Menurut *Olson Delen* (2008) menjelaskan *Naive Bayes* untuk setiap kelas keputusan, menghitung probabilitas dengan syarat bahwa kelas keputusan adalah benar, mengingat vektor informasi objek. Algoritma ini mengasumsikan bahwa atribut objek adalah independen. Probabilitas yang terlibat dalam memproduksi perkiraan akhir dihitung sebagai jumlah frekuensi dari **Master** tabel keputusan. *Naive Bayes Classifier* bekerja sangat baik dibanding dengan model classifier lainnya.

Keuntungan penggunaan adalah bahwa metode ini hanya membutuhkan jumlah data pelatihan (*training data*) yang kecil untuk menentukan estimasi parameter yang diperlukan dalam proses pengklasifikasian. Karena yang di asumsikan sebagai *variable independent*, maka hanya varians dari suatu *variable* dalam sebuah kelas yang dibutuhkan untuk menentukan klasifikasi, bukan keseluruhan dari matriks kovarians (Rayuwati et al., 2022).

Menurut (Rayuwati et al., 2022), Algoritma *Naïve Bayes* memiliki beberapa kegunaan, kelebihan, kekurangan, serta ada persamaan Metode *Naïve Bayes* dan alur dari metode *Naïve Bayes*.

2.3.1. Kelebihan dan Kekurangan *Naive Bayes*

1. Kelebihan *Naive Bayes*

   Naive Bayes bisa dipakai untuk data kuantitatif maupun kualitatif, Tidak memerlukan jumlah data yang banyak, Tidak perlu melakukan data training yang banyak, Jika ada nilai yang hilang, maka bisa diabaikan dalam perhitungan, Pengklasifikasian

dokumen bisa dipersonalisasi, Jika digunakan dalam bahasa pemrograman, code-nya sederhana dan Bisa digunakan untuk klasifikasi masalah biner ataupun multiclass. Selain kelebihan diatas, *Naive Bayes* memiliki beberapa kegunaan, yaitu:

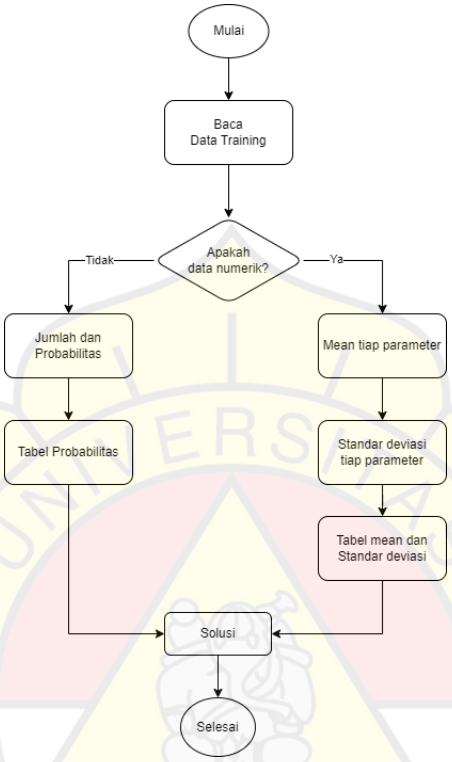
- Mengklasifikasikan dokumen teks seperti teks berita ataupun teks akademis
- Sebagai metode machine learning yang menggunakan probabilitas
- Untuk membuat Prioritas Ticket secara otomatis

2. Kekurangan *Naive Bayes*

   Apabila probabilitas *Naive Bayes* ini kondisional nya bernilai nol, maka probabilitas prediksi juga akan bernilai nol, asumsi bahwa masing-masing variabel independen membuat berkurangnya akurasi, Keakuratannya tidak bisa diukur menggunakan satu probabilitas saja, dan untuk membuat keputusan, diperlukan pengetahuan awal atau pengetahuan mengenai masa sebelumnya. Keberhasilannya sangat bergantung pada pengetahuan awal tersebut Banyak celah yang bisa mengurangi efektivitas nya dirancang untuk mendeteksi kata-kata saja, tidak bisa berupa gambar

2.3.2. Formula *Naive Bayes*

1. Alur dari metode *Naive Bayes*



Gambar 2. 1 Alur Metode Naive Bayes

Keterangan gambar 2.1:

1. Baca data training
2. Hitung Jumlah dan probabilitas, namun apabila data numerik maka
    1. Cari nilai mean dan standar deviasi dari masing-masing parameter yang merupakan data numerik
    2. Adapun persamaan yang digunakan untuk menghitung nilai rata – rata hitung (mean) dapat dilihat sebagai berikut:

$$\mu = \frac{\sum_{i=1}^{n} x_i}{n}$$ atau $$\mu = \frac{x_1 + x_2 + x_3 + \ldots + x_n}{n}$$

Dimana:

μ : rata –rata hitung(mean)

Xi : nilai sample ke - i

N : Jumlah sampel

3. Mendapatkan nilai dalam tabel mean, standard deviasi dan probabilitas.
4. Solusi kemudian dihasilkan

2. Rumus dari metode *Naive Bayes*

$$P(H|X) = \frac{P(X|H).P(H)}{P(X)}$$

Dimana:

X	:	*Data dengan class yang belum diketahui*
H	:	*Hipotesis data merupakan suatu class spesifik.*
P(H|X)	:	*Probabilitas hipotesis H berdasarkan kondisi X (posteriori probabilitas).*
P(H)	:	*Probabilitas hipotesis H (prior probabilitas).*
P(X|H)	:	*Probabilitas X berdasarkan kondisi pada hipotesis H*
P(X)	:	*Probabilitas X*

Untuk menjelaskan metode *Naive Bayes*, perlu diketahui bahwa proses klasifikasi memerlukan sejumlah petunjuk untuk menentukan kelas apa yang cocok bagi sampel yang dianalisis tersebut. Karena itu, metode *Naive Bayes* di atas disesuaikan:

$$P(C|F1 \ldots Fn) = \frac{P(C)P(F1 \ldots Fn|C)}{P(F1 \ldots Fn)}$$

Di mana Variabel C merepresentasikan kelas, sementara variabel F1 ... Fn merepresentasikan karakteristik petunjuk yang dibutuhkan untuk melakukan klasifikasi. Maka rumus tersebut menjelaskan

bahwa peluang masuknya sampel karakteristik tertentu dalam kelas C (Posterior) adalah peluang munculnya kelas C (sebelum masuknya sampel tersebut, seringkali disebut prior), dikali dengan peluang kemunculan karakteristik-karakteristik sampel pada kelas C (disebut juga likelihood), dibagi dengan peluang kemunculan karakteristik-karakteristik sampel secara global (disebut juga evidence). Nilai *Evidence* selalu tetap untuk setiap kelas pada satu sampel. Nilai dari *posterior* tersebut nantinya akan dibandingkan dengan nilai-nilai *posterior* kelas lainnya untuk menentukan ke kelas apa suatu sampel akan diklasifikasikan.

2.4 Konsep Dasar Aplikasi

2.4.1. Pengertian Aplikasi

Aplikasi adalah program yang dibuat dengan tujuan untuk melaksanakan fungsi sesuai dengan kegunaan aplikasinya, penggunaannya, dan jenis aplikasi itu sendiri. Aplikasi dibuat dengan bahasa pemrograman yang bertujuan untuk membantu memecahkan masalah dengan aturan yang sesuai dengan bahasa pemrograman itu sendiri yang nantinya bisa mengolah data (Pane et al., 2020).

Pengertian aplikasi secara umum adalah alat terapan yang difungsikan secara khusus dan terpadu sesuai kemampuan yang dimilikinya aplikasi merupakan suatu perangkat komputer yang siap pakai bagi *user* (Siregar, H. F., Siregar, Y. H., & Melani, 2018).

Berdasarkan kedua penyataan diatas dapat disimpulkan bahwa Aplikasi adalah program komputer yang dibuat untuk melakukan suatu tugas

tertentu. Aplikasi dapat dibuat dengan berbagai bahasa pemrograman, dan dapat digunakan untuk berbagai keperluan, seperti bisnis, pendidikan, hiburan, dan lain-lain.

2.4.2. Pengertian *Framework*

Framework atau dalam bahasa indonesia dapat diartikan sebagai "kerangka kerja" merupakan kumpulan dari fungsi-fungsi/prosedur-prosedur dan class-class untuk tujuan tertentu yang sudah siap digunakan sehingga bisa lebih mempermudah dan mempercepat pekerjaan seorang programer, tanpa harus membuat fungsi atau class dari awal (Bin Tahir et al., 2019).

2.4.3. Pengertian *Laravel*

Aplikasi adalah program komputer yang dibuat untuk melakukan suatu tugas tertentu. Aplikasi dapat dibuat dengan berbagai bahasa pemrograman, dan dapat digunakan untuk berbagai keperluan, seperti bisnis, pendidikan, hiburan, dan lain-lain (Bin Tahir et al., 2019).

Dalam website (M Ali Maksum, 2022) pada jurnal (Amarulloh, 2023) di katakan, *Laravel* adalah *Framework* berbasis bahasa pemrograman *PHP* yang bisa digunakan untuk membantu proses pengembangan sebuah website agar lebih maksimal. Dengan menggunakan *Laravel*, website yang dihasilkan akan lebih dinamis.

Berdasarkan kedua penyataan diatas dapat disimpulkan bahwa *Laravel* adalah sebuah *Framework* bahasa pemrograman *PHP* yang memiliki banyak fitur modern yang sangat membantu para pengembang dalam pembuatan aplikasi dan pengembangan website. Beberapa

keunggulan *Laravel* meliputi penggunaan *Command Line Interface* (CLI) Artisan, penggunaan *package manager PHP Composer*, penulisan kode program yang lebih singkat, mudah dimengerti, dan ekspresif. Pada Aplikasi *Helpdesk* ini, saya menggunakan beberapa package untuk membantu kinerja dari aplikasi tersebut, diantaranya:

1. yooper/*PHP*-text-analysis

   Library yooper/*PHP*-text-analysis adalah sebuah library *PHP* yang menyediakan berbagai fungsi untuk melakukan analisis teks. Library ini dapat digunakan untuk melakukan berbagai tugas, seperti klasifikasi dokumen, analisis sentimen, dan pengenalan pola. Fungsi-fungsi yang tersedia dalam library yooper/*PHP*-text-analysis antara lain:

   a. **Klasifikasi Dokumen**: Fungsi ini dapat digunakan untuk mengklasifikasi dokumen berdasarkan kontennya. Misalnya, dokumen dapat diklasifikasikan berdasarkan topik, genre, atau emosi.
   b. **Analisis Sentimen**: Fungsi ini dapat digunakan untuk menentukan sentimen dari teks, apakah positif, negatif, atau netral.
   c. **Pengenalan Pola**: Fungsi ini dapat digunakan untuk mengenali pola dalam teks, seperti frasa, kata, atau karakter.

2. cviebrock/eloquent-sluggable

   eloquent-sluggable adalah sebuah library *PHP* yang menyediakan fungsi untuk membuat slug untuk model Eloquent. Slug adalah

string unik yang digunakan untuk mewakili model Eloquent. Slug biasanya digunakan untuk URL, nama file, atau nama lain yang perlu unik.

3. *Laravel*-Excel

   Laravel-Excel adalah sebuah library *PHP* yang menyediakan fungsi-fungsi untuk melakukan ekspor dan impor data Excel dari *Laravel*. Library ini dibangun di atas library *PHP*Spreadsheet, yang merupakan library *PHP* untuk membaca dan menulis file Excel. Fungsi-fungsi yang tersedia dalam library antara lain:

   a. **Ekspor Data**: Library ini menyediakan berbagai fungsi untuk mengekspor data dari *Laravel* ke file Excel. Data dapat diekspor dari berbagai sumber, seperti model Eloquent, koleksi, atau data yang dihasilkan dari query *SQL*.

   b. **Impor Data**: Library ini menyediakan berbagai fungsi untuk mengimpor data dari file Excel ke *Laravel*. Data dapat diimpor ke berbagai sumber, seperti model Eloquent, koleksi, atau *Database*.

4. *OpenAI-PHP*/client

   ***OpenAI-PHP*/client** adalah sebuah library *PHP* yang menyediakan antarmuka pemrograman aplikasi (API) untuk berinteraksi dengan *OpenAI* API. *OpenAI* API adalah API yang memungkinkan pengembang untuk menggunakan model kecerdasan buatan (*AI*) *OpenAI*, seperti GPT-3, untuk berbagai tugas. Library ini

menyediakan berbagai fungsi untuk mengakses *OpenAI* API, termasuk:

a. Text generation: Fungsi ini dapat digunakan untuk menghasilkan teks, seperti puisi, kode, skrip, karya musik, email, surat, dll.
b. Translation: Fungsi ini dapat digunakan untuk menerjemahkan teks dari satu bahasa ke bahasa lain.
c. Answering questions: Fungsi ini dapat digunakan untuk menjawab pertanyaan dengan cara yang informatif.
d. Categorizing text: Fungsi ini dapat digunakan untuk mengklasifikasikan teks berdasarkan topik, genre, atau emosi.
e. Content creation: Fungsi ini dapat digunakan untuk membuat konten, seperti artikel, blog, atau situs web.

2.4.4. Pengertian *Livewire*

Livewire merupakan library yang dibuat untuk *Framework PHP* tertentu, yaitu *Laravel*. *Livewire* memiliki fitur yang menarik yaitu real-time. Kecepatan pemrosesan input *Livewire* berbeda dari kerangka kerja lain. Untuk memproses input, biasanya setelah memasukkan data, frame lain akan memuat ulang browser untuk menghapus data sebelumnya. Dan ini berarti sangat berguna untuk memasukkan data dalam jumlah besar dan menghemat waktu (Daru & Adhiwibowo, 2021).

2.4.5. *Database* (Basis Data)

Database adalah kumpulan informasi yang disimpan secara sistematis di dalam komputer sehingga dapat dikendalikan oleh program komputer untuk mengambil informasi dari *Database*. Istilah "basis data" berasal dari ilmu komputer. Kemudian diperluas untuk memasukkan hal-hal selain elektronik. Catatan seperti *Database* ada sebelum Revolusi Industri dalam bentuk buku, kuitansi, dan kumpulan data bisnis (Andaru, 2018).

Database adalah suatu susunan atau kumpulan catatan data yang tersimpan di dalam komputer. Hubungan antar entri dalam *Database* dapat digunakan sebagai sumber informasi bagi pengguna. Sampai saat ini, masih banyak record *Database* yang ditampilkan dalam bentuk teks sebagai informasi kepada pengguna. Ini adalah salah satu kerentanan yang dimiliki analis kriptografi dalam mengakses, memanipulasi atau membocorkan dan mendistribusikan catatan basis data (Yanti et al., 2018).

Berdasarkan kedua penyataan diatas dapat disimpulkan bahwa *Database* adalah kumpulan data yang disimpan secara sistematis di dalam komputer dan dapat diakses oleh program komputer untuk mengambil data dari *Database* tersebut. Saat ini, banyak record *Database* masih ditampilkan dalam bentuk teks, yang merupakan ancaman keamanan data karena orang yang tidak berwenang dapat mengakses, mengubah, membocorkan, dan menyebarkan data. Oleh

karena itu, analis kriptografi sangat penting dalam menangani keamanan dan kerahasiaan data basis data.

A. *SQL*

SQL merupakan bahasa terstruktur yang khusus digunakan untuk mengolah *Database*. *SQL* juga dapat diartikan sebagai antar muka standar untuk sistem manajemen relasional, termasuk sistem yang beroperasi pada komputer pribadi. *SQL* lebih mudah digunakan dibandingkan dengan bahasa pemrograman, tetapi rumit dibandingkan *software* lembar kerja dan pengolah data (Novendri et al., 2019).

SQL (*Structured Query Language*) adalah sebuah konsep pengoperasian *Database* terutama untuk pemilihan atau seleksi dan pemasukan data yang memungkinkan pengoperasian data dikerjakan dengan mudah secara otomatis (Warman & Ramdaniansyah, 2018).

B. My*SQL*

MySQL merupakan *software* sistem manajemen *Database* (*Database Management System/DBMS*) yang sangat popular dikalangan pemrogram. Kepopuleran *MySQL* dimungkinkan karena kemudahannya untuk digunakan, cepat secara kinerja query dan mencukupi untuk kebutuhan *Database* perusahaan-perusahaan skala menengah kecil (Rejeki et al., 2021).

MySQL adalah Relational *Database Management System* (*RDBMS*) yang didistribusikan secara gratis dibawah lisensi *GPL (General Public License)*. Dimana setiap orang bebas untuk

menggunakan *MySQL*, namun tidak boleh dijadikan produk turunan yang bersifat komersil (Warman & Ramdaniansyah, 2018).

2.4.6. Pengertian *Laragon*

Laragon ialah perangkat lunak bebas yang di dalamnya terdapat banyak sistem operasi sebagai localhost atau server mandiri. *Laragon* menyediakan banyak layanan, peralatan, dan fitur yang terdiri dari Apache, *PHP* Server, *PHP*MyAdmin, My*SQL*, Memchaced, Redis, Composer, Xdebug, Cmder dan *Laravel* (Putra et al., 2019).

2.5 *Helpdesk*

Helpdesk merupakan sistem yang digunakan untuk memberikan bantuan, dukungan, dan solusi terhadap berbagai masalah atau pertanyaan yang muncul di SMK Pelita Alam. *Customer Helpdesk system plays an important role in assisting the end users or customers of the organization to get the resolutions for their service-related problems.*(S.P. & K.S., 2021). Pada Artikel Jurnal (S.P. & K.S., 2021), Paramesh dan Shreedhara membahas tentang pengklasifikasian tiket secara otomatis dengan menggunakan konsep kecerdasan buatan (*Artificial Intelligence*) seperti *Text Document Classification* and *Natural Language Processing Technique.* Dikatakan bahwa sistem *Helpdesk* berhasil jika mereka dapat menangani dan menyelesaikan masalah dengan cepat dan akurat untuk setiap keluhan yang disampaikan ke departemen yang relevan dalam hal IT (MIS). Di kutip dari (Tarigan et al., 2022), tugas dari *MIS* adalah pelayanan terhadap pengaduan/keluhan dari *user*

mengenai masalah kerusakan *hardware*, *software* dan jaringan dengan menerapkan sistem IT *Helpdesk.*

2.5.1. Kelebihan *Helpdesk*

Helpdesk dapat menyediakan pelayanan yang terbaik kepada para karyawan serta pelanggan juga mengurangi biaya pada organisasi yang dapat memberikan produktifitas. Hal ini disebabkan karena kelebihan-kelebihan yang ada pada *Helpdesk*, antara lain (Ambo & Winoto, 2021) :

1. *Helpdesk* dapat memberikan solusi atas pertanyaan-pertanyaan maupun keluhan yang masuk dalam waktu yang lebih singkat.
2. *Helpdesk* dapat mengecek status permasalahan yang ada dan mengatur pembagian kerja staff.
3. *Helpdesk* dapat meningkatkan efisiensi perusahaan dalam menangani pertanyaan dan keluhan dari pelanggan.
4. *Helpdesk* dapat memberikan laporan kerja perkembangan kinerja para staff kepada pemimpin.

2.6 e-*Ticketing*

Menurut (Muda et al., 2021), e-*Ticketing* adalah suatu cara untuk mendokumentasikan proses penjualan online yang dapat memudahkan calon pembeli untuk dalam pemesanan tiket lewat web application.

2.6.1. Cara Kerja e-*Ticketing*

Menurut (Tarigan et al., 2022), e-*Ticketing* memungkinkan *user* melihat secara real time mengenai status perkembangan dari penanganan masalah yang mereka hadapi. R. Tarigan, Dkk juga

menjelaskan bahwa Sistem e-*Ticketing* dapat mempermudah pembuatan tiket secara elektronik dan pelaporan juga dapat dilakukan secara elektronik.

2.7 *Chatbot*

Conversational Bot atau *Chatbot* adalah sebuah konten yang divisualisasikan dalam format obrolan dan pengguna dapat berinteraksi dengan sistem menggunakan teks (Yuniar & Purnomo, 2019).

Menurut Eka Yuniar dan Heri Purnomo, *Chatbot* telah dibekali dengan kecerdasan buatan dan pemrosesan bahasa alami atau *NLP* yang membuatnya menjadi aplikasi komputer yang cerdas dan dapat menjawab pertanyaan yang diberikan oleh manusia, *Chatbot* ini dibangun dengan menerapkan sistem pakar dengan menggunakan metode *forward chaining*. Metode *forward chaining* digunakan untuk mencari kesimpulan dari fakta-fakta yang terkumpul.

2.8 *UML* (*Unified Model Language*)

Menurut (Kurniawan et al., 2021), *UML* atau *Unified Model Language* merupakan standar bahasa dalam dokumentasi, melakukan spesifikasi, dan pembangunan dalam pengembangan sebuah *software*, yang mana menggunakan pendekatan *Object Oriented Programming.*

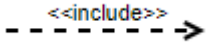
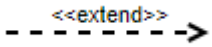
5.1 Use Case Diagram

Use Case Diagram merupakan diagram atau gambaran grafik dari beberapa aktor dan bagaimana mereka berinteraksi satu sama lain. Ini menunjukkan sistem, siapa yang dapat menjalankan prosedur pada sistem, dan proses yang terkait. Selama tahap pemodelan,

pengembang aplikasi sering menggunakan *Use Case* karena dapat dilihat hubungan antar aktor dan sistem dapat melakukan apa saja (Kurniawan et al., 2021).

Tabel 2. 1 Use Case Diagram (Hendini, 2016)

Simbol	Nama	Keterangan
	*Use Case*	*Use Case* menggambarkan fungsionalitas yang disediakan sistem sebagai unit-unit yang bertukar pesan antar unit dengan aktor, yang dinyatakan dengan menggunakan kata kerja
	Aktor	Actor atau Aktor adalah Abstraction dari orang atau sistem yang lain yang mengaktifkan fungsi dari target sistem. Untuk mengidentifikasikan aktor, harus ditentukan pembagian tenaga kerja dan tugas-tugas yang berkaitan dengan peran pada konteks target sistem. Orang atau sistem bisa muncul dalam beberapa peran. Perlu dicatat bahwa aktor berinteraksi dengan *Use Case*, tetapi tidak memiliki kontrol terhadap *Use Case*
	*Association*	Asosiasi antara aktor dan *Use Case* yang menggunakan panah terbuka untuk mengindikasikan bila aktor berinteraksi secara pasif dengan sistem.

	*Include*	*Include*, merupakan di dalam *Use Case* lain (required) atau pemanggilan *Use Case* oleh *Use Case* lain, contohnya adalah pemanggilan sebuah fungsi program
	*Extend*	*extend*, merupakan perluasan dari *Use Case* lain jika kondisi atau syarat terpenuhi

5.2 Activity Diagram

Aktivitas diagram merupakan rancangan aliran aktivitas atau kerja dalam sistem yang akan dijalankan. Selain itu, aluran tampilan sistem dapat digambarkan dalam diagram aktivitas. Aktivitas diagram memiliki komponen yang dihubungkan dengan tanda panah. Panah tersebut menghasilkan rangkaian aktivitas yang berlangsung dari awal hingga akhir (Rizky, 2019).

Tabel 2. 2 Activity Diagram (Rizky, 2019)

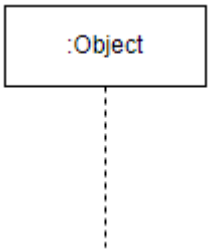
Simbol	Nama	Keterangan
	*Start Point*	*Start Point*, diletakkan pada pojok kiri atas dan merupakan awal aktivitas
	*End Point*	*End Point*, akhir aktivitas
	*Activities*	*Activities*, menggambarkan suatu proses / kegiatan bisnis

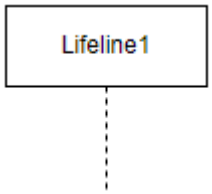
	Join	*Join* (penggabungan) atau *rake*, digunakan untuk menunjukkan adanya dekomposisi
	*Decision Points*	*Decision Points*, menggambarkan pilihan untuk pengambilan keputusan, *true* atau *false*
	*Swimlane*	*Swimlane*, pembagian *Activity diagram* untuk menunjukkan siapa melakukan apa

5.3 Sequence Diagram

Sequence Diagram menggambarkan kelakuan objek pada *Use Case* dengan mendeskripsikan waktu hidup objek dan pesan yang dikirimkan dan diterima antar objek. (Hendini, 2016) menjelaskan Simbol-simbol yang digunakan dalam *Sequence Diagram* yaitu:

Tabel 2. 3 Sequence Diagram

Simbol	Nama	Keterangan
	*Aktor*	Menggambarkan seorang pengguna yang sedang berinteraksi dengan sistem.
	*Object*	*Object* berguna untuk mendokumentasikan perilaku sebuah *object* pada sebuah sistem.

	*Lifeline*	*Lifeline*, garis titik-titik yang terhubung dengan objek, sepanjang *lifeline* terdapat *activation*
	*Activation Bar*	*Activation*, mewakili sebuah eksekusi operasi dari objek, panjang kotak ini berbanding lurus dengan durasi aktivasi sebuah operasi
	*Recursive*	*Recursive*, menggambarkan pengiriman pesan yang dikirim untuk dirinya sendiri

5.4 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.

Class Diagram secara khas meliputi: Kelas (*Class*), Relasi *Associations*, *Generalization* dan *Aggregation*, attribut (*Attributes*), operasi (*operation/method*) dan *visibility*, tingkat akses objek eksternal kepada suatu operasi atau attribut. Hubungan antar kelas

mempunyai keterangan yang disebut dengan *Multiplicity* atau *Cardinality* (Hendini, 2016).

Tabel 2. 4 Class Diagram

Multiplicity	Penjelasan
1	Satu Instansi
0..1	Nol atau Satu Instansi
0..*	Nol atau lebih dari Satu Instansi
1..*	Satu atau lebih dari Satu Instansi

BAB III

TEKNOLOGI INFORMASI

UNIVERSITAS DARMA PERSADA