BAB II

LANDASAN TEORI

2.1. Konsep Dasar

2.1.1. Implementasi

Secara sederhana implementasi bisa diartikan pelaksanaan atau penerapan. Browne dan Wildavsky (Usman, 2004:7) mengemukakan bahwa "implementasi adalah perluasan aktivitas yang saling menyesuaikan".

Menurut Syaukani dkk (2004: 295) implementasi merupakan suatu rangkaian aktivitas dalam rangka menghantarkan kebijakan kepada masyarakat sehingga kebijakan tersebut dapat membawa hasil sebagaimana diharapkan. Rangkaian kegiatan tersebut mencakup, Pertama persiapan seperangkat peraturan lanjutan yang merupakan interpretasi dari kebijakan tersebut. Kedua, menyiapkan sumber daya guna menggerakkan kegiatan implementasi termasuk didalamnya sarana dan prasarana, sumber daya keuangan dan tentu saja penetapan siapa yang bertanggung jawab melaksanakan kebijaksanaan tersebut. Ketiga, bagaimana mengahantarkan kebijaksanaan secara kongkrit ke masyarakat.

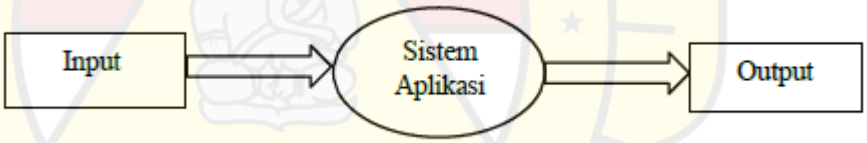
2.1.2. Sistem

Sistem adalah Sekumpulan elemen atau unsur yang saling berhubungan untuk mencapai tujuan tertentu. Hal ini juga sependapat oleh teori yang di sampaikan oleh Tohari dalam (Faizal

& Putri, 2017). Menjelaskan bahwa “Sistem adalah kumpulan atau himpunan dari unsur atau variable-variabel yang saling terkait, saling berinteraksi, dan saling tergantung satu sama lain, untuk mencapai suatu tujuan”.

Sedangkan menurut Gelinas dan Dull dalam (Faizal & Putri, 2017). Menjelaskan bahwa “Sistem adalah seperangkat elemen independent yang bersama-sama mencapai tujuan yang spesifik”. Dari pengertian tersebut, dapat di simpulkan bahwa sistem adalah seperangkat atau kumpulan dari unsur atau variable yang saling terkait dan berinteraksi untuk mencapai tujuan tertentu.

Dalam sebuah Sistem terdapat beberapa komponen dasar serta karakteristik yang mendukung suatu sistem tersebut



Hubungan antar elemen-elemen yang terdapat dalam sistem menurut Andri Kristanto (2008:2), meliputi:

1. Tujuan Sistem

Sistem yang dibuat harus memiliki tujuan (*Goal*). Sistem bisa memiliki hanya satu tujuan namun juga bisa memiliki lebih dari satu tujuan. Tujuan inilah yang menjadi pemotivasi yang mengarahkan sistem. Tanpa tujuan, sistem menjadi tak terarah dan tak terkendali.

2. **Input (Masukkan)**

Masukan (*input*) sistem adalah segala sesuatu yang masuk ke dalam sistem dan selanjutnya menjadi bahan yang diproses. Masukan dapat berupa hal- hal yang berwujud (tampak secara fisik) maupun yang tidak tampak.

3. **Output (Keluaran)**

Merupakan hasil dari *input* yang telah diproses oleh bagian pengolahan dan merupakan tujuan akhir sistem. *Output* dapat berupa informasi berguna yang dapat ditangkap oleh indera manusia, semisal berupa cetakan laporan dan informasi.

2.1.3. Peramalan

Menurut Assauri (2016:73), prakiraan ramalan biasanya diklasifikasikan atas cakupan lamanya atau horizon waktu ke depan. Metode peramalan sendiri memiliki dua kategori utama yaitu metode kuantitatif dan metode kualitatif. Peramalan Kuantitatif yaitu memperkirakan secara kuantatif mengenai apa yang terjadi dimasa yang akan datang Metode peramalan kuantitatif dapat dibagi menjadi dua bagian yaitu metode peramalan deret waktu dan metode kausal, sedangkan metode kualitatif dibagi menjadi metode eksploratoris dan normatif. Teknik peramalan kuantitatif sangat beragam, dikembangkan dari berbagai disiplin ilmu dan untuk berbagai maksud. Setiap teknik yang akan dipilih memiliki sifat, ketepatan, tingkat kesulitan dan biaya tersendiri yang harus dipertimbangkan. (Dirpan,2007), menjelaskan bahwa pada

umumnya peramalan kuantitatif dapat diterapkan bila terdapat tiga kondisi berikut:

1. Tersedia informasi tentang masa lalu (data historis)
2. Informasi tersebut dapat dikuantitatifkan dalam bentuk numerik.
3. Dapat diasumsikan bahwa beberapa aspek pola masa lalu akan terusberlanjut di masa mendatang.

Umumnya horizon waktu ke depan dibedakan atas tiga kategori, yaitu:

1. Prakiraan ramalan jangka pendek, yang mencakup jarak waktu dari tiga bulan sampa dengan dengan satu tahun. Perkiraan ramalan jangka pendek ini digunakan dalam penyusunan rencana pembelian, penjadwalan tugas pekerjaan atau *job shceduling*, penetapan level tenaga kerja atau *workforce levels*, pemberian tugas (*job assigments*), dan tingkat produksi (*production levels*).
2. Prakiraan ramalan jangka menengah (*median range forecast*), umumnya prakiraan ramalan ini mencakup masa waktu dari satu tahun sampai dengan tiga tahun. Prakiraan ramalan jangka menengah ini digunakan dalam penyusunan rencana penjualan, perencanaan produksi dan budgeting atau penganggaran yang meliputi anggaran kas, dan analisis berbagai rencana produksi.
3. Prakiraan ramalan jangka panjang, umumnya prakiraan ramalan ini mencakup masa waktu tiga tahun atau lebih. Prakiraan ramalan jangka panjang ini digunakan untuk perencanaan produk

baru, anggaran pengeluaran modal atau capital expenditure, perencanaan lokasi fasilitas ekspansi, dan riset & pengembangan (*Research & Development*).

2.1.4. Pengertian Persediaan

Pengertian persediaan menurut Sofyan Assauri dalam buku Marihot Manullang dan Dearlina Sinaga (2005:50), menerangkan bahwa Persediaan adalah sebagai suatu aktiva lancar yang meliputi barang – barang milik perusahaan dengan maksud untuk dijual dalam suatu periode usaha normal atau persediaan barang – barang yang masih dalam pekerjaan proses produksi ataupun persediaan bahan baku yang menunggu penggunaanya dalam suatu proses produksi.

Menurut Zaki Badridwan (2000:149), menerangkan bahwa Pengertian persediaan barang secara umum istilah persediaan barang dipakai untuk menunjukan barang-barang yang dimiliki untuk dijual kembali atau digunakan untuk memproduksi barang-barang yang akan dijual.

2.1.5. Pengertian *Data Mining*

Menurut Abdul Kadir (2007:30), *Data mining* adalah kombinasi secara logis antara pengetahuan data, dan analisa statistik yang dikembangkan dalam pengetahuan bisnis atau suatu proses yang menggunakan teknik statistik, matematika, kecerdasan buatan, tiruan dan *machine-learning* untuk mengekstraksi dan

mengidentifikasi informasi yang bermanfaat bagi pengetahuan yang terkait dari berbagai *database* besar.

Pada dasarnya *data mining* berhubungan erat dengan analisis datadan penggunaan perangkat lunak untuk mencari pola dan kesamaan dalam sekumpulan data. Ide dasarnya adalah menggali sumber yang berharga dari suatutempat yang sama sekali tidak diduga, seperti perangkat lunak *data mining* mengekstrasi pola yang sebelumnya tidak terlihat atau tidak begitu jelas sehingga tidak seorang pun yang memperhatikan sebelumnya. Analisa *data mining* berjalan pada data yang cenderung terus membesar dan teknik terbaik yang digunakan kemudian berorientasi kepada data berukuran sangat besar untuk mendapatkan kesimpulan dan keputusan paling layak. *Data mining* memiliki beberapa sebutan atau nama lain yaitu: *knowledge discovery in database* (KDD), ekstraksi pengetahuan (*knowledge extraction*), analisa data / pola (data / *pattern analysis*), kecerdasan bisnis (*business intelligence*), data *archaeology* dan data *dredging* (Larose, 2005).

2.1.6. Metode *Data Mining*

Secara garis besar, Han dalam bukunya menjelaskan bahwa metode data mining dapat dilihat dari dua sudut pandang pendekatan yang berbeda, yaitu pendekatan deskriptif dan pendekatan prediktif [8]. Pendekatan deskriptif adalah pendekatan

dengan cara mendeskripsikan data inputan. Metode yang termasuk ke dalam pendekatan ini adalah:

1. Metode deskripsi konsep/kelas, yaitu data dapat diasosiasikan dengan kelas atau konsep. Ada tiga macam pendeskripsian yaitu (1) karakteristik data, dengan membuat summary karakter umum atau fitur data suatu kelas target, (2) diskriminasi data, dengan membandingkan *class* target dengan satu atau sekelompok kelas pembanding, (3) gabungkan antara karakterisasi dan diskriminasi.
2. Metode *association rule*, yaitu menemukan aturan asosiatif atau pola kombinasi dari suatu item yang sering terjadi dalam sebuah data. Pendekatan kedua adalah pendekatan prediktif, yaitu pendekatan yang dapat digunakan untuk memprediksi, dengan hasil berupa kelas atau *cluster*.
3. Metode klasifikasi dan prediksi, yaitu metode analisis data yang digunakan untuk membentuk model yang mendeskripsikan kelas datayang penting, atau model yang memprediksikan trend data. Klasifikasi digunakan untuk memprediksi kelas data yang bersifat kategorial, sedangkan prediksi untuk memodelkan fungsi yang mempunyai nilai kontinu.
4. Metode *clustering*, mengelompokkan data untuk membentuk kelas- kelas baru atau sering disebut *cluster*. Metode *clustering* bertujuan untuk memaksimalkan persamaan dalam satu *cluster* dan meminimalkan perbedaan antar *cluster*.

2.1.7. Metode *Association Rule*

Salah satu pengaplikasian dari *Association Rule* adalah *Market Basket Analysis*, dimana bertujuan untuk menemukan bagaimana item yang dibeli oleh pelanggan dalam toko saling berhubungan. Pencarian *Association Rules* dilakukan melalui dua tahap yaitu pencarian *frequent itemset* dan penyusunan *rules*. Penting tidaknya suatu *Association Rules* dapat diketahui dengan dua parameter, yaitu *support* (nilai penunjang) dan *confidence* (nilai kepastian). *Support* adalah ukuran yang menunjukan tingkat dominasi itemset dari kesuluruhan transaksi. Persamaan I untuk menentukan nilai suatu *support*

Ada 2 (dua) proses utama dalam penggalian aturan asosiasi, yaitu: pencarian pola (*frequent pattern*) dari sejumlah transaksi penentuan kuatnya *rule* (aturan) dari pola yang dihasilkan. Dalam *association rule mining* ada 2 (dua) hal yang mempengaruhi:

1. *Support*

   Support adalah proporsi suatu item dalam semua transaksi. *Support* dirumuskan sebagai berikut:

$$supp(x) = \frac{\sum x}{\sum transaksi}$$

2. *Confidence*

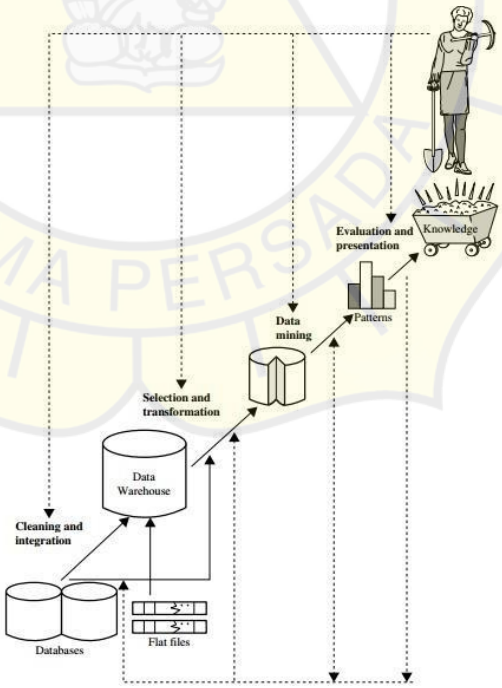
   Confidence adalah hal yang mendasari aturan asosiasi, dengan konsep implikasi (x => y), atau "*if ... then ...*". Besarnya nilai c*onfidence* suatu aturan (*rule*) dirumuskan sebagai berikut:

$$conf(x => y) = \frac{\sum xy}{\sum x}$$

Nilai minimum *support* dan minimum *confidence* bisa diatur oleh pengguna ataupun ahli yang berkaitan (Han, Jiawei. 2012).

2.1.8. Tahapan *Data Mining*

Menururut Syaifullah (2010:15), Dalam aplikasinya, data mining sebenarnya merupakan bagian dari proses *Knowledge Discovery in Database* (KDD), bukan sebagai teknologi yang utuh dan berdiri sendiri. *Data mining* merupakan suatu bagian langkah yang penting dalam proses KDD terutama berkaitan dengan ekstraksi dan dari data yang ditelaah, seperti ditunjukanoleh gambar 1 dibawah ini



Gambar 2. 1 *Knowledge discovery from data*

1. *Data Selection*

   Pemilihan (seleksi) data dari sekumpulan data operasional perlu dilakukan sebelum tahap penggalian informasi dalam KDD dimulai. Data hasil seleksi yang kan digunakan untuk proses data mining, disimpan dalam suatu berkas, terpisah dari basis data operasional.

2. *Pre-processing/Cleaning*

   Sebelum proses *data mining* dapat dilaksanakan, perlu dilakukan proses *cleaning* pada data yang menjadi focus KKD. Proses *cleaning* mencakup antara lain membuang duplikasi data, memeriksa data yang inkonsisten, dan memperbaiki kesalahan pada data, seperti kesalahan cetak (tipografi). Juga dilakukan proses *enrichment*, yaitu "memperkaya" data yang sudah ada dengan data informasi atau informasi lain yang relevan dan diperlukan untuk

   a. Transformasi

      Coding adalah proses transformasi pada data yang telah dipilih, sehingga data tersebut sesuai untuk proses data mining. Proses coding dalam KKD merupakan proses kreatif dan tergantung pada jenis atau pola informasi yang akan dicari dalam basis data.

3. *Data Mining*

   Data mining adalah proses mencari pola atau informasi menarik dalam data terpilih dengan menggunakan teknik atau metode tertentu. Teknik, metode atau algoritma dalam data mining sangat

bervariasi. Pemilihan metode atau algoritma yang tepat sangat bergantung pada tujuan dan proses KKD secara keseluruhan.

4. *Interpretation/Evaluation*

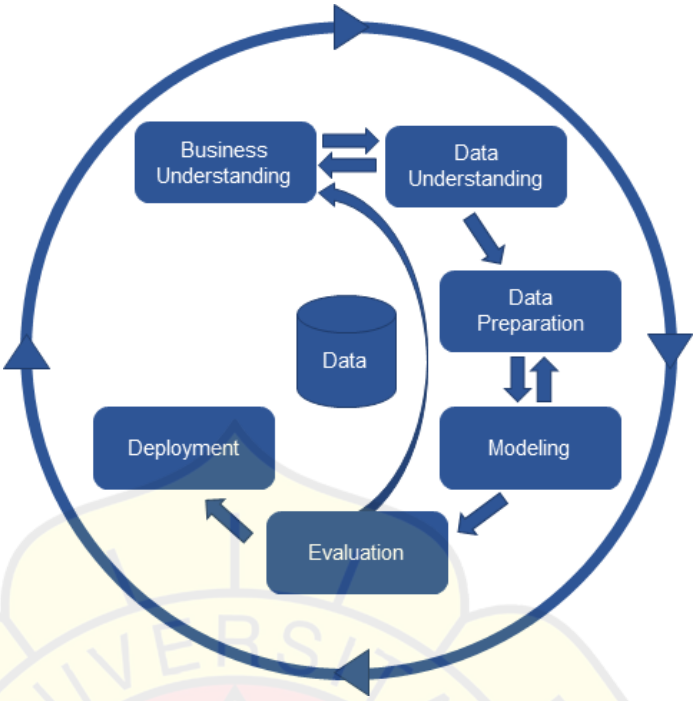
   Pola informasi yang dihasilkan dari proses data mining perlu ditampilkan dalam bentuk yang mudah dimengerti oleh pihak yang berkepentingan. Tahap ini merupakan bagian dari proses KKD yang disebut dengan interpretation. Tahap ini mencakup pemeriksaan apakah pola atau informasi yang ditemukan bertentangan dengan fakta atau hipotesa yang ada sebelumnya.

2.1.9. *Cross Industry Standard Process for Data Mining*

Cross Industry Standard for Data Mining (CRIS – DM) yang dikembangkan tahun 1996 oleh analis dari beberapa industry seperti Daimbler Chrysler, SPSS, dan NCR. CRISP DM menyediakan standar proses data mining sebagai strategi pemecahan masalah secara umum dari bisnis atau unit penelitian.

Dalam CRISP – DM, sebuah proyek *data mining* memiliki siklus hidup yang terbagi dalam enam fase. Keseluruhan fase berurutan yang ada tersebut bersifat adaptif. Fase berikutnya dalam urutan bergantung kepada keluaran dari fase sebelumnya.

Hubungan penting antarfase digambarkan dengan panah. Sebagai contoh, jika proses berada pada fase modelling. Berdasar pada perilaku dan karakteristik model, proses mungkin harus kembali kepada fase data *preparation* untuk perbaikan lebih lanjut terhadap data atau berpindah maju kepada fase *evaluation*.



Gambar 2. 2 *CRISP-DM*

Enam fase CRISP – DM (Larose, 2005):

1. Fase Pemahaman Bisnis (*Business Understanding Phase*)
    a. Penentuan tujuan objek dan kebutuhan secara detail dalam lingkup bisnis atau unit penelitian secara keseluruhan.
    b. Menerjemahkan tujuan dan batasan menjadi formula dari permasalahan *data mining*.
    c. Menyiapkan strategi awal untuk mencapai tujuan.
2. Fase Pemahaman Data (*Data Understanding Phase*)
    a. Mengumpulkan data.
    b. Menggunakan analisis penyelidikan data untuk mengenali lebih lanjut data dan pencarian pengetahuan awal.
    c. Mengevaluasi kualitas data.

d. Jika diinginkan, pilih sebagian kecil grup data yang mungkin mengandung pola dari permasalahan.

3. Fase Pengolahan Data (*Data Preparation Phase*)
   a. Siapkan dari data awal, kumpulkan data yang ingin digunakan untuk keseluruhan fase berikutnya. Fase ini merupakan pekerjaan berat yang perlu dilaksanakan secara intensif.
   b. Pilih kasus dan variabel yang ingin dianalisis dan yang sesuai analisis yang akan dilakukan.
   c. Lakukan perubahan pada beberapa variabel jika dibutuhkan.
   d. Siapkan data awal sehingga siap untuk perangkat pemodelan.
4. Fase Pemodelan (*Modelling Phase*)
   a. Pilih dan aplikasikan teknik pemodelan yang sesuai.
   b. Kalibrasi aturan model untuk mengoptimalkan hasil.
   c. Perlu diperhatikan bahwa beberapa teknik mungkin untuk digunakan pada permasalahan data mining yang sama.
   d. Jika diperlukan, proses dapat kembal ke fase pengolahan data untuk menjadikan data ke dalam bentuk yang sesuai dengan spesifikasi kebutuhan teknik *data mining* tertentu.
5. Fase Evaluasi (*Evaluation Phase*)
   a. Mengevaluasi satu atau lebih model yang digunakan dalam fase pemodelan untuk mendapatkan kualitas dan efektifitas sebelum disebarkan untuk digunakan.

b. Menetapkan apakah terdapat model yang memenuhi tujuan pada fase awal.

c. Menentukan apakah terdapat permasalahan penting dari bisnis atau penelitian yang tidak tertangani dengan baik.

d. Mengambil keputusan berkaitan dengan penggunaan hasil dari *data mining*.

6. Fase Penyebaran (*Deployment Phase*)

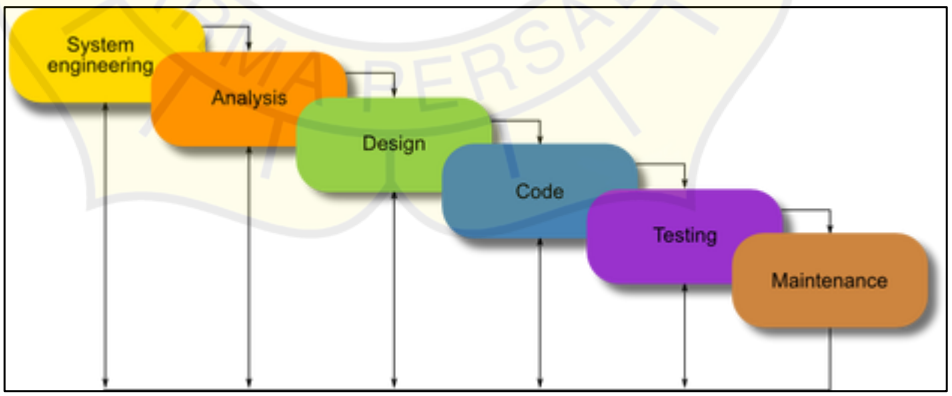
a. Menggunakan model yang dihasilkan. Terbentuknya model tidak menandakan telah terselesaikannya proyek.

b. Contoh sederhana penyebaran: Pembuatan laporan.

c. Contoh kompleks penyebaran: Penerapan proses data mining secara paralel pada departemen lain.

2.2. Metode *Waterfall*

Dalam pengembangan sistem ini penulis menggunakan Metode *Waterfall*.



Gambar 2. 3 Ilustrasi model *waterfall*

Menurut Rosa A.S & M. Shalahuddin (2018, h.28 & h.29), Model SDLC air terjun (*waterfall*) sering juga disebut medel sekuensial linier (*sequential*

linear) atau alur hidup klasik (*classic life cycle*). Model *waterfall* menyediakan pendekatan alur hidup perangkat lunak secara sekuensial atau terurut dimulai dari analisa, desain, pengodean, pengujian, dan tahap pendukung (*support*).

f. Analisis Kebutuhan

Proses pengumpulan kebutuhan dilakukan secara intensif untuk menspesifikasikan kebutuhan perangkat lunak agar dapat dipahami perangkat lunak seperti apa yang dibutuhkan oleh *user*. Spesifikasi kebutuhan perangkat lunak pada tahap ini perlu untuk didokumentasikan.

g. Desain

Desain perangkat lunak adalah proses multi langkah yang focus pada desain pembuatan program perangkat lunak termasuk struktur data, arsitektur perangkat lunak, representasi antarmuka, dan prosedur pengodean. Tahap ini mentranslasi kebutuhan perangkat lunak dari tahap analilis kebutuhan ke representasi desain agar dapat diimplementasikan menjadi program pada tahap selanjutnya. Desain perangkat lunak yang dihasilkan pada tahap ini juga perlu didokumentasikan.

h. Pembuatan kode program

Desain harus ditranslasikan ke dalam program perangkat lunak. Hasil dari tahap ini adalah program komputer sesuai dengan desain yang telah dibuat pada tahap desain.

i. Pengujian

Pengujian fokus pada perangkat lunak secara dari segi lojik dan fungsional dan memastikan bahwa semua bagian sudah diuji. Hal ini dilakukan untuk meminimalisir kesalahan (*error*) dan memastikan keluaran yang dihasilkan sesuai dengan yang diinginkan.

j. Pendukung (*support*) atau pemeliharaan (*maintenance*)

Tidak menutup kemungkinan sebuah perangkat lunak mengalami perubahan ketika sudah dikirimkan ke *user*. Perubahan bias terjadi karena adanya kesalahan yang muncul dan tidak terdeteksi saat pengujian atau perangkat lunak harus beradaptasi dengan lingkungan baru. Tahap pendukung atau pemeliharaan dapat mengulangi proses pengembangan mulai dari analisi spesifikasi untuk perubahan perangkat lunak yang sudah ada, tapi tidak untuk membuat perangkat lunak baru.

2.3. Metode Peramalan

2.3.1. Teori *Exponential Smoothing*

Exponential Smoothing (penghalusan *exponential*) (Santoso, 2009) adalah salah satu tipe teknik peramalan rata-rata bergerak yang melakukan penimbangan terhadap data masa lalu dengan cara eksponensial sehingga data paling akhir mempunyai bobot atau timbangan lebih besar dalam rata-rata bergerak.

Metode *Exponential Smoothing* (Makridakis, 1999) merupakan prosedur perbaikan terus-menerus pada peramalan terhadap objek pengamatan terbaru. Metode peramalan ini menitik-

beratkan pada penurunan prioritas secara eksponensial pada objek pengamatan yang lebih tua.

Smoothing adalah mengambil rata – rata dari nilai pada beberapa periode untuk menaksir nilai pada suatu periode (Pangestu Subagyo, 1986:7), *Exponential Smoothing* adalah suatu metode peramalan rata- rata bergerak yang melakukan pembobotan menurun secara *exponential* terhadap nilainilai observasi yang lebih tua (Makridakis,1993:79)

Pemulusan eksponensial atau *exponential smoothing* terdapat satu atau lebih parameter pemulusan yang ditentukan secara eksplisit, dan hasil ini menentukan bobot yang dikenakan pada nilai observasi, dengan kata lain observasi terbaru akan diberikan prioritas lebih tinggi bagi peramalan daripada observasi yang lebih lama. Metode *exponential smoothing* merupakan pengembangan dari metode *moving average*. Dalam metode ini peramalan dilakukan dengan mengulang perhitungan secara terus menerus dengan menggunakan data terbaru. Metode ini dibagi menjadi tiga, yaitu *single exponential smoothing*, *double exponential smoothing* dan *triple exponential smoothing*. Model ramalan *exponential smoothing* merupakan salah satu model ramalan data berkala (*time series*).

Beberapa keunggulan metode penghalusan eksponensial (*exponential smoothing*) dibandingkan dengan metode tradisional (Leabo Dick A., 1968:322) adalah:

1. Data-data selalu dioperasikan dengan efisien
2. Hanya membutuhkan sedikit data dari satu waktu ke waktu berikutnya
3. Dapat dimodifikasi untuk mengolah data yang berisi trend tertentu atau pola musiman
4. Dapat digunakan dengan biaya murah baik secara manual maupun dengan komputer.

2.3.2. Metode *Single Exponential Smoothing*

Single exponential smoothing yang digunakan pada peramalan jangka pendek, biasanya hanya 1 bulan ke depan. Model mengasumsikan bahwa data berfluktuasi di sekitar nilai mean yang tetap, tanpa trend atau pola pertumbuhan konsisten. (Makridakis, 1999). Rumus untuk *Simple exponential smoothing* adalah sebagai berikut:

$$F_{t+1} = \propto X_t + (1 - \propto) F_t \text{----------(2.1)}$$

Dimana :

F_t = peramalan untuk periode t

$X_t + (1 - \propto)$ = Nilai aktual time series

(F_{t+1}) = peramalan pada waktu t + 1

$\propto$ = konstanta perataan antara 0 dan 1

Metode *single exponential smoothing* merupakan perkembangan dari metode moving average sederhana, yang mula–mula dengan rumus sebagai berikut :

$$S_{1+t} = \frac{X_1 + X_{t-1} + \cdots + X_{t-n+1}}{n}$$ ……..…… …(2.2)

$$S_t = \frac{X_{t-1} + X_{t-2} + \cdots + X_{t-n}}{n}$$ ……..…… …..(2.3)

Dengan melihat hubungan diatas bila S_t diketahui maka nilai S_{t+1} dapat dicari berdasarkan St.

$$S_{t+1} = \frac{X_t}{n} + S_t - \frac{X_{t-n}}{n}$$

…….……..…….(2.4)

Bila $\frac{X_{t-n}}{n}$ diganti dengan nilai peramalan pada t yaiti St maka persamaan menjadi:

$$S_{t+1} = \frac{X_t}{n} + S_t - \frac{S_t}{n}$$

…….……..(2.5)

Atau

$$S_{t+1} = \frac{1}{n} X_t + \left(1 - \frac{1}{n}\right) S_t$$

…….……(2.6)

$$\frac{1}{n} = \propto$$

Sehingga persamaannya menjadi:

$$S_{t+1} = \propto X_t + (1 - \propto) S_t$$

…….……...(2.7)

Penjelasan :

S_{t+1} = Ramalan untuk periode ke t+1

X_t = Nilai forecast riil pada periode ke-t

α= Konstanta bobot yang menunjukkan konstanta penghalus (nilai antara 0sampai 1)

S_t = Nilai ramalan forecast pada periode ke-t

2.3.3. Algoritma *FP-Growth*

Algoritma *FP-Growth* merupakan pengembangan dari algoritma Apriori. Algoritma *Frequent Pattern Growth* (*FP-Growth*) adalah salah satu alternatif algoritma yang dapat digunakan untuk menentukan himpunan data yang paling sering muncul (*frequent itemset*) dalam sebuah kumpulan data. *FP-Growth* dapat menemukan frekuensi itemset dengan hanya sedikit mengakses pada *database* aslinya, dan pendekatannya adalah yang paling efisien. (Wiyana,2018)

Algoritma *FP-Growth* juga dapat menghindari permasalahan jika jumlah calon itemset nya terlalu besar. *FP-Growth* menggunakan konsep pembangunan *tree* dalam pencarian *frequent itemset*. Hal tersebut yang menyebabkan algoritma *FP-Growth* lebih cepat dari algoritma Apriori. Dengan menggunakan *FP-Tree*, algoritma *FP-Growth* dapat langsung mengekstrak

frequent itemset dari *FP-Tree*. Penggalian itemset yang frequent dengan menggunakan algoritma *FP-Growth* akan dilakukan dengan cara membangkitkan struktur data *tree*. (Erwin, 2009)

2.3.4. Pembangunan *FP-Tree*

FP-Tree merupakan struktur penyimpanan data yang dimampatkan. *FP- tree* dibangun dengan memetakan setiap data transaksi ke dalam setiap lintasan tertentu dalam *FP-tree*. Karena dalam setiap transaksi yang dipetakan mungkin ada transaksi yang memiliki item yang sama, maka lintasannya memungkinkan untuk saling menimpa. Semakin banyak data transaksi yang memiliki item yang sama, maka proses pemampatan dengan struktur data *FP-tree* semakin efektif. Kelebihan dari *FP-tree* adalah hanya memerlukan dua kali pemindaian data transaksi yang terbukti sangat efisien. *FP-tree* adalah sebuah pohon dengan defenisi sebagai berikut.

1. *FP-tree* dibentuk oleh sebuah akar yang diberi label null, sekumpulan upapohon yang beranggotakan item-item tertentu, dan sebuah *table frequent header*.
2. Setiap simpul dalam *FP-tree* mengandung tiga informasi penting yaitu label item menginformasikan jenis item yang dipresentasikan simpul tersebut, dan pointer penghubung yang menghubungkan simpul-simpul dengan label item sama antar lintasan, ditandai dengan garis panah putus-putus. (Wijaya, 2018).

Tahapan dalam pembentukan *FP-tree* adalah sebagai berikut: (Samuel, 2008)

1. Tentukan data transaksi dengan *minimum support count* =2

Tabel 2. 1 Table Data Transaksi Awal

TID	Transaksi
1	a,b
2	b,c,d,g,h
3	a,c,d,e,f
4	a,d,e
5	a,b,z,c
6	a,b,c,d
7	a,r
8	a,b,c
9	a,b,d
10	b,c,e

Frekuensi kemunculan tiap item dapat dilihat pada table berikut:

Tabel 2. 2 Frekuensi Kemunculaan Tiap Karakter

Item	Frekuensi
a	8
b	7
c	6
d	5

e	3
f	1
r	1
z	1
g	1
h	1

Frekuensi kemunculan tiap item dapat dilihat pada table berikut:

Tabel 2. 3 Frekuensi Kemunculaan Tiap Karakter

Item	Frekuensi
a	8
b	7
c	6
d	5
e	3
f	1
r	1
z	1
g	1
h	1

Setelah dilakukan pemindaian pertama didapat item yang memiliki frekuensi di atas support count□=2 adalah a,b,c,d dan e. kelima item inilah yang akan berpengaruh dan akan

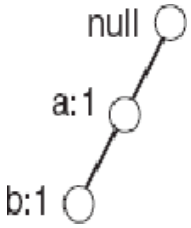
dimasukkan kedalan *FP-tree*, selebihnya r,z,g,h dapat dibuang karena tidak berpengaruh signifikan.

Tabel berikut mendata kemunculan item yang frequent dalam setiap transaksi, diurut berdasarkan yang frekuensinya paling tinggi.

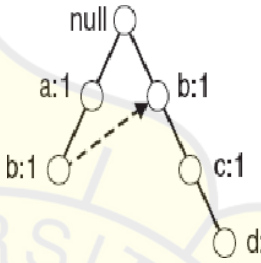
Tabel 2. 4 Table Data Transaksi

TID	Transaksi
1	{a,b}
2	{b,c,d}
3	{a,c,d,e}
4	{a,d,e}
5	{a,b,c}
6	{a,b,c,d}
7	{a}
8	{a,b,c}
9	{a,b,d}
10	{b,c,e}

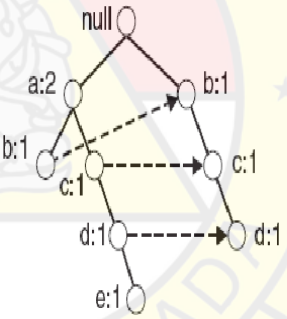
Gambar dibawah ini adalah ilustrasi pembentukan *FP-tree* setelah pembacaan TID 1.



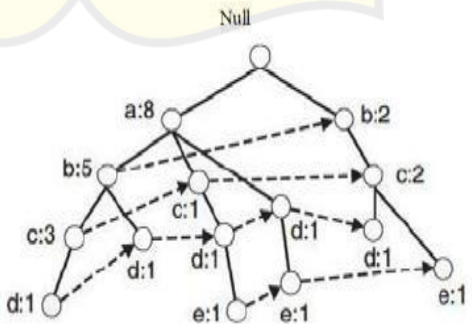
Gambar 2. 4 Hasil pembentukan *FP-tree* setelah pembacaan TID 1



Gambar 2. 5 Hasil pembentukan *FP-tree* setelah pembacaan TID 2



Gambar 2. 6 Hasil pembentukan *FP-tree* setelah pembacaan TID 3



Gambar 2. 7 Hasil pembentukan *FP-tree* setelah pembacaan TID 10

Diberikan 10 data transaksi dengan 5 jenis item seperti pada tabel di atas. Gambar 1 – 4 menunjukkan proses terbentuknya *FP-tree* setiap TID dibaca. Setiap simpul pada *FP-tree* mengandung nama sebuah item dan counter support yang berfungsi untuk menghitung frekuensi kemunculan item tersebut dalam tiap lintasan transaksi. (Arsyad, 2013)

FP-tree yang merepresentasikan data transaksi pada tabel 2.1 dibentuk dengan cara sebagai berikut: (Samuel, 2008)

1. Kumpulan data dipindai pertama kali untuk menentukan support count dari setiap item. Item yang tidak *frequent* dibuang, sedangkan *frequent* item dimasukkan dan disusun dengan urutan menurun, seperti yang terlihat pada tabel 2.1.
2. Pemindaian kedua, yaitu pembacaan TID pertama {a, b} akan membuat simpul a dan b, sehingga terbentuk lintasan transaksi Null→a→b. Support count dari setiap simpul bernilai awal 1
3. Setelah pembacaan transaksi kedua {b, c, d}, terbentuk lintasan kedua yaitu Null→b→c→d. *Support count* masing-masing count juga bernilai awal 1. Walaupun b ada pada transaksi pertama, namun karena prefix transaksinya tidak

sama, maka transaksi kedua ini tidak bisa dimampatkan dalam satu lintasan.

4. Transaksi keempat memiliki prefix transaksi yang sama dengan transaksi pertama, yaitu a, maka lintasan transaksi ketiga dapat ditimpakan di a, sambil menambah *support count* dari a, dan selanjutnya membuat lintasan baru sesuai dengan transaksi ketiga.
5. Proses ini dilanjutkan sampai *FP-tree* berhasil dibangun berdasarkan tabel data transaksi yang diberikan.

Algoritma *FP-Growth* memiliki tahapan-tahapan yang harus dilewati agar dapat memberikan hasil yang maksimal, tahapan-tahapan tersebut, yaitu: (Anggraeni, 2014)

1. Tahap pembangkitan *conditional pattern base*

   Pembangkitan *conditional pattern base* merupakan *sub database* yang berisi *prefix path* dan *suffix pattern*. Pembangkitan *conditinal pattern base* didapatkan melalui *FP-Tree* yang telah dibangun sebelumnya.
2. Tahap pembangkitan *conditional FP-Tree*

   Pada tahap ini, *support count* dari setiap item pada setiap *conditional pattern base* dijumlahkan, lalu setiap item yang memiliki jumlah *support count* lebih besar atau sama dengan *minimum support count* yang akan dibangkitkan dengan *conditional FP-Tree*.
3. Tahap pencarian *frequent itemset.*

Apabila conditional *FP-Tree* merupakan lintasan tunggal (*single path*), maka didapatkan *frequent pattern* dengan melakukan kombinasi item untuk setiap *conditional FP-Tree*. Jika bukan lintasan tunggal, maka dilakukan pembangkitan *FP-Growth* secara rekursif.

Ketiga tahap tersebut merupakan langkah yang akan dilakukan untuk mendapatkan *frequent itemset*. Setelah memeriksa frequent itemset untuk beberapa akhiran (*suffix*), maka didapat hasil yang dirangkum dalam tabel berikut:

Tabel 2. 5 Hasil *Frequent Itemset*

Suffix	***Frequent Itemset***
e	{e},{d,e},{a,d,e},{c,e},{a,e}
d	{d},{c,d},{b,c,d},{a,c,d},{b,d},{a,b,d},{a,d}
c	{c},{b,c},{a,b,c},{a,c}
b	{b},{a,b}
a	{a}

Dengan metode *divide and conquer* ini, maka pada setiap langkah rekursif, algoritma *FP-growth* akan membangun sebuah *conditional FP-tree* baru yang telah diperbaharui nilai *support count*, dan membuang lintasan yang mengandung item-item yang tidak frequent lagi. (Astrina, 2019).

2.4. Pemrograman Aplikasi

2.4.1. *Web*

Menurut Ilka Zufria dan M. Hasan Azhari (2017:52), *Website* adalah sekumpulan halaman informasi yang disediakan melalui jalur internet sehingga bisa diakses di seluruh dunia selama terkoneksi dengan jaringan internet.

Menurut Triyono (2018:23), *Website* atau situs dapat diartikan sebagai kumpulan halaman-halaman yang digunakan untuk menampilkan informasi teks, gambar diam atau bergerak, animasi, suara, dan atau gabungan dari semuanya baik yang bersifat dinamis yang membentuk satu rangkaian bangunan yang saling terkait, yang masing-masing dihubungkan dengan jaringan-jaringan halaman.

Berdasarkan pengertian para ahli diatas dapat disimpulkan bahwa *website* adalah kumpulan dari keseluruhan halaman-halaman *web* yang berisi sebuah data atau informasi baik yang bersifat statis maupun dinamis yang membentuk satu rangkaian bangunan yang saling terkait, yang masing-masing dihubungkan dengan jaringan-jaringan halaman.

2.4.2. HTML

Hypertext Markup Language (HTML) merupakan bahasa dasar *web* yang berfungsi untuk menampilkan berbagai komponen *web*. HTML dikembangkan pertama kali oleh Tim Berners. Menurut Jurnal yang di tulis oleh Achmad Solichin (2019) tujuan

utama pengembangan HTML adalah untuk menghubungkan satu halaman *web* dengan halam *web* lainya. Dengan kata lain HTML adalah fondasi *web*. HTML disusun dengan bahasa yang sederhana, sehingga sangat mudah diimplementasikan.

Kode HTML yang dibuat nantinya akan diterjemahkan *web browser*supaya bisa tampil seperti apa yang sudah dirancang. Sebenarnya, semua *web browser* bisa menampilkan kode HTML dengan baik, akan tetapi jika berbicara tentang desain halaman, maka setip *browser* tentu memiliki beberapa perbedaan.

HTML memang dirancang serta diatur badan standarisasi dunia khusus yang menangani *web* yakni *World Wide Web Consortium* [W3C]. Ini disebabkan karena masing masing program *web browser* akan menerjemahkan kode HTML dengan berbeda sehingga dibutuhkan standar yang sama untuk semua *browser*.

2.4.3. CSS

Cascading Style Sheets (CSS) merupakan salah satu kode pemrograman yang bertujuan untuk menghias dan mengatur gaya tampilan halaman *web* supaya lebih elegan dan menarik. CSS adalah teknologi internet yang direkomendasikan *world wide web consortium* (W3C) pada tahun 1996. CSS juga digunakan oleh *web programmer* dan juga *web designer* untuk menentukan warna, tata letak font, dan semua aspek lain dari presentasi dokumen disitus mereka menurut Jurnal yang di tulis oleh (Didik Setiawan, 2018).

Ada dua sifat CSS yaitu internal dan eksternal. Jika internal yang dipilih, maka skrip itu dimasukkan secara langsung ke halaman *website* yang akan didesain. Kalau halaman *web* yang lain akan didesain dengan model yang sama, maka skrip CSS itu harus dimasukkan lagi ke dalam halaman *web* yang lain itu. Sifat yang kedua adalah eksternal dimana skrip CSS dipisahkan dan diletakkan dalam berkas khusus. Nanti cukup gunakan semacam tautan menuju berkas CSS itu jika halaman *web* yang didesain akan dibuat seperti model yang ada di skrip tersebut.

2.4.4. *JavaScript*

Menurut Jurnal yang di tulis oleh (R.H. Sianipar, 2018) *javascript* adalah bahasa *scripting* yang popular di internet dan dapat bekerja di sebagian besar *browser* popular seperti *Internet Explorer*, *Mozilla Firefox*, *Netscape* dan *Opera Mini*. Kode *javascript* dapat disisipkan dalam halaman *web* menggunakan tag *script*. Berikut ini beberapa sifat dari *javascript*:

- Menambahkan interaktiktivitas ke halaman HTML.
- Merupakan bahasa pemrograman *scripting*.
- Bahasa *Scripting* merupakan bahasa yang ringan.
- *Javascript* merupakan bahasa terinterpretasi.

2.4.5. *Bootstrap*

Bootstrap adalah *framework front-end* yang intuitif dan powerful untuk pengembangan aplikasi *web* yang lebih cepat dan mudah. *Bootstrap* menggunakan HTML, CSS, dan *Javasript*.

Bootstrap memilki fitur-fitur komponen *interface* yang bagus seperti *Typografi, Forms, Buttons, Navigations, Dropdowns, Alerts, Modals, Tabs, Accordian, Carousel,* dan lain sebagainya. Dengan demikian dalam membuat *website* kita bisa menghemat waktu, fitur yang *responsive*, dan memiliki *design* yang konsisten menurut Jurnal yang di tulis oleh (Gregorius Agung, 2018).

Bootstrap telah menyediakan kumpulan aturan dan komponen *class interface* dasar sebagai modal dalam pembuatan *web* yang telah dirancang sangat baik untuk memberikan tampilan yang sangat menarik, bersih, ringan dan memudahkan bagi penggunanya. Dan penggunaan *bootstrap* ini kita juga diberikan keleluasan salama pengembangan *website*, anda bisa merubah dan menambah *class* sesuai dengan keinginan. *Bootstrap* memberikan kemudahan bagi anda, dengan menggunakannya dapat memangkas waktu, tenaga dalam proses pengerjaan suatu *website*. Kita selalu dituntut melakukan pekerjaan apapun dengan efisien dan efektif, dengan demikian penggunaan *framework twitter bootstrap* ini bisa anda pilih ketika membuat suatu *website* bagi anda maupun klien anda.

2.4.6. PHP

Supono dan Putratama mengemukakan bahwa PHP (PHP: *Hypertext Preprocessor*) adalah suatu bahasa pemrograman yang digunakan untuk menerjemahkan baris kode program menjadi kode mesin yang dapat dimengerti oleh komputer yang berbasis *server-*

side yang dapat ditambahkan ke dalam HTML menurut jurnal yang di tulis oleh (Saputra, Agus. 2018).

2.4.7. *Database*

Pengertian *Database* Menurut (Hesananda et al, 2017), *Database* ialah suatu wadah untuk menampung sebuah data yang ada pada sebuah sistem. *Database* juga bias diartikan sebagai kumpulan data. *Database* juga biasa dikenal formal dan tegas. *Database* juga bias diartikan dengan kumpulan data yang terintegrasi yang dapat dimanipulasi, diambil dan dicari secara cepat.

Jadi secara umum Basis data (*database*) adalah kumpulan informasi yang disimpan didalam komputer secara sistematik sehingga dapat diperiksa menggunakan suatu program komputeruntuk memperoleh informasi dari basis data tersebut. Perangkat lunak yang digunakan untukmengelola dan memanggil kueri (*query*) basis data tersebut disebut sistem manajemen basis data (*Database Management System*).

2.4.8. MySQL

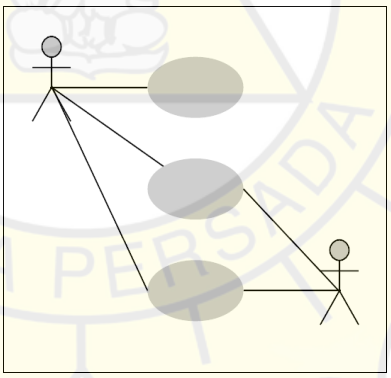
Menurut Jurnal yang di tulis oleh (R.H. Sianipar, 2018) dalam Buku "Membangun *Web* dengan PHP & MYSQL untuk Pemula & Programmer". MySQL bukan termasuk bahasa pemrograman. MySQL merupakan salah satu *database* popular dan mendunia. MySQL bekerja menggunakan SQL Language (*Structure Query Language*). Pada umumnya, perintah yang paling

sering digunakan dalam MySQL adalah *SELECT* (mengambil), *INSERT* (menambah), *UPDATE* (mengubah), dan *DELETE* (menghapus). Selain itu, SQL juga menyediakan perintah untuk membuat *database*, *field*, ataupun *index* untuk menambah atau menghapus data jurnal (Sianipar, R.H. 2015).

2.5. Pemodelan UML

Unified Modelling Language (UML) adalah sebuah bahasa yang telah menjadi standar dalam industri untuk evaluasi, merancang, dan mendokumentasikan sistem peranti lunak. UML menawarkan sebuah standar untuk merancang model sebuah sistem. UML juga dapat digunakan untuk aplikasi *modeling procedural* seperti VB atau C. jurnal (Yuni Sugiarti, 2018).

2.5.1. *Usecase* Diagram

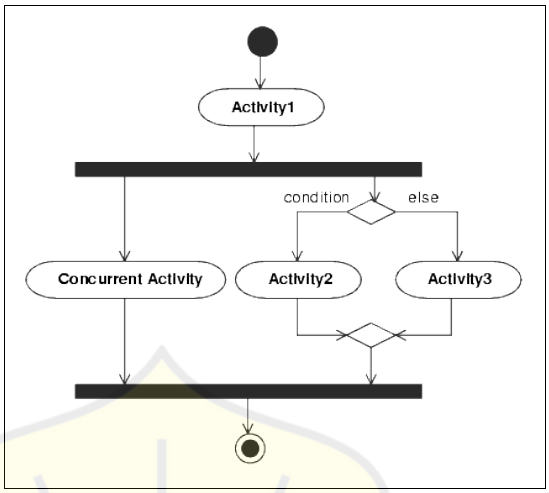


Gambar 2. 8 *Use Case* Diagram

Sumber: Jurnal (Yuni Sugiarti, 2018)

UseCase diagram merupakan pemodelan untuk menggambarkan *behavior* dan mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem yang akan dibuat. Jurnal (Yuni Sugiarti,2018).

2.5.2. *Activity* Diagram

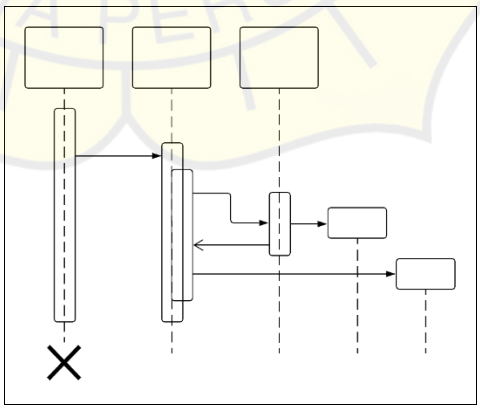


Gambar 2. 9 *Activity* Diagram

Sumber: Jurnal (Yuni Sugiarti,2018)

Activity diagram menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis. Hal yang perlu diperhatikan di sini adalah bahwa diagram aktivitas menggambarkan kegiatan sistem bukan apa yang dilakukan aktor, jadi aktivitas yang dapat dilakukan oleh sistem. (Yuni Sugiarti,2018)

2.5.3. *Sequence* Diagram



Gambar 2. 10 *Sequence* Diagram

Sumber Jurnal (Yuni Sugiarti,2018)

Diagram sekuens (*sequence)* menggambarkan *behavior* objek pada *Usecase* dengan mendeskripsikan waktu hidup dan *message* yang dikirimkan dan diterima antarobjek. Banyaknya diagram sekuens yang harus digambar adalah sebanyak pendefinisian *Usecase* yang memiliki proses sendiri atau yang penting semua *Usecase* telah didefinisikan interaksi jalannya pesan sudah dicakup pada diagram sekuens. (Yuni Sugiarti,2018)