BAB 2

Landasan Teori

Bab ini dijelaskan beberapa konsep dan dasar teori yang berkaitan dengan permasalahan yang akan dibahas oleh penyusun sebagai dasar pemahaman dalam mengimplementasikan konsep-konsep tersebut ke dalam semua kegiatan pengembangan sistem

2.1 Peramalan (*Forecasting*)

2.1.1 Pengertian Peramalan (*Forecasting*)

Menurut Sri Ramadhani Harahap (2021), Peramalan (*forecasting*) adalah kegiatan memperkirakan atau memprediksi apa yang akan terjadi di masa yang akan datang dengan waktu yang relatif lama. Sedangkan ramalan adalah suatu situasi atau kondisi yang diperkirakan akan terjadi di masa yang akan datang. Untuk memprediksi hal tersebut diperlukan data yang akurat di masa lalu, sehingga dapat dilihat prospek situasi dan kondisi di masa yang akan datang.

Menurut Reynold Sitorus (2019), Peramalan bertujuan untuk memperkirakan prospek ekonomi dan kegiatan usaha serta pengaruh lingkungan terhadap prospek tersebut. Dalam kondisi pasar bebas, permintaan pasar lebih banyak bersifat kompleks dan dinamis karena permintaan tersebut akan tergantung pada keadaan sosial, ekonomi, sosial politik, aspek teknologi, produk pesaing dan produk substitusi. Oleh karena itu peramalan yang akurat merupakan informasi yang sangat dibutuhkan dalam pengambilan keputusan manajemen.

Peramalan adalah suatu taksiran atau perkiraan nilai-nilai sebuah variable berdasarkan kepada nilai yang diketahui dari variable tersebut atau variable yang berhubungan dengan bantuan perhitungan statistik dalam memperoleh gambaran kejadian dimasa mendatang.

Peramalan adalah suatu proses memperkirakan secara sistematis tentang apa yang mungkin terjadi dimasa yang akan datang berdasarkan masa lalu dan sekarang yang dimiliki agar kesalahannya dapat diperkecil. Peramalan tidak memberikan jawaban pasti tentang apa yang akan terjadi, melainkan berusaha mencari pendekatan tentang apa yang terjadi sehingga dapat memberikan kontribusi dalam menentukan keputusan yang terbaik.

2.1.2 Fungsi dan Tujuan Peramalan

Fungsi peramalan atau *forecasting* terlihat pada saat pengambilan keputusan. Keputusan yang baik adalah keputusan yang didasarkan atas pertimbangan apa yang akan terjadi pada waktu keputusan itu dilaksanakan. Apabila kurang tepat ramalan yang disusun, maka masalah peramalan juga merupakan masalah yang selalu dihadapi (Rahamneh, Adeeb Ahmed Ali Al 2017).

Menurut Heizer dan Render (2009), peramalan atau *forecasting* memiliki tujuan sebagai berikut:

1. Untuk mengkaji kebijakan perusahaan yang berlaku saat ini dan di masa lalu serta melihat sejauh mana pengaruh di masa datang.
2. Peramalan diperlukan karena adanya time lag atau delay antara saat suatu kebijakan perusahaan ditetapkan dengan saat implementasi.

3. Peramalan merupakan dasar penyusunan bisnis pada suatu perusahaan sehingga dapat meningkatkan efektivitas suatu rencana bisnis.

2.1.3 Jenis – Jenis Peramalan

Menurut Render dan Heizer (2001:47) jenis peramalan dapat dibedakan menjadi beberapa tipe. Dilihat dari perencanaan operasi di masa depan, maka peramalan dibagi menjadi 3 macam yaitu:

1. Peramalan ekonomi (*economic forecast*) menjelaskan siklus bisnis dengan memprediksi tingkat inflasi, ketersediaan uang, dana yang dibutuhkan untuk membangun perumahan dan indikator perencanaan lainnya.
2. Peramalan teknologi (*technological forecast*) memperhatikan tingkat kemajuan teknologi yang dapat meluncurkan produk baru yang menarik, yang membutuhkan pabrik dan peralatan yang baru.
3. Peramalan permintaan (*demand forecast*) adalah proyeksi permintaan untuk produk atau layanan perusahaan. Proyeksi permintaan untuk produk atau layanan suatu perusahaan. Peramalan ini juga disebut peramalan penjualan yang mengendalikan produksi, kapasitas, serta sistem penjadwalan dan menjadi input bagi perencanaan keuangan, pemasaran, dan sumber daya manusia.

2.2 Metodelogi Sistem

2.2.1 Metode Single Moving Average (SMA)

Single Moving Average adalah salah satu metode peramalan Time series (deret waktu). Metode ini digunakan jika data masa lalu tidak memiliki unsur trend atau faktor musiman. Tujuan dilakukannya peramalan rata - rata bergerak tunggal adalah untuk menghilangkan atau mengurangi acakan (*random ness*) dalam deret waktu. Tujuan ini dapat dicapai dengan merata - ratakan beberapa nilai dalam data bersama–sama, dengan cara mana kesalahan *positif* dan *negatif* yang mungkin terjadi dan dapat dikeluarkan atau dihilangkan.

Single Moving Average adalah suatu metode peramalan yang dilakukan dengan mengambil sekelompok nilai pengamatan, mencari nilai rata - rata tersebut sebagai ramalan untuk periode yang akan datang (Nurul Hudaningsih, 2020). Menurut (Ni Luh Ayu K Y,2014) Metode ini mempunyai dua sifat khusus yaitu untuk membuat forecast memerlukan data historis dalam jangka waktu tertentu, semakin panjang moving average akan menghasilkan moving *averages* yang semakin halus, secara sistematis moving *average* adalah.

$$St+1 = (n1 + n2 + n3 + \ldots) / n$$

Gambar 2.1 Formula Sma

Dimana :

St + 1 = Forecast untuk period ke t+1.

n1 = Data pada periode t.

n = Jangka waktu Moving *averages*.

nilai n merupakan banyaknya periode dalam rata-rata bergerak.

2.2.2 Metode Double Exponential Smoothing (DES)

Metode *Double Exponential Smoothing* adalah suatu metode yang paling luas digunakan untuk menentukan persamaan trend data pemulusan kedua melalui proses *smoothing*. *Double Exponential Smoothing* digunakan untuk data yang memiliki trend atau data yang memiliki kecenderungan peningkatan atau penurunan dalam jangka panjang. Sistem peramalan ini menangkap pola dari data yang telah lalu kemudian digunakan untuk memproyeksikan data yang akan datang.

Metode *Double Exponential Smoothing* merupakan model linear yang dikemukakan oleh *Brown*. Dalam metode ini dilakukan proses smoothing dua kali. Dasar pemikiran metode pemulusan *eksponensial linear* dari *Brown* adalah serupa dengan rata-rata bergerak linear, karena kedua nilai pemulusan tunggal dan ganda ketinggalan dari data yang sebenarnya jika terdapat unsur trend. Perbedaan antara nilai pemulusan tunggal dan ganda dapat ditambahkan dengan nilai pemulusan tunggal dan disesuaikan untuk trend. Persamaan yang dipakai dalam implementasi pemulusan eksponensial linear satu-parameter dari *Brown* adalah sebagai berikut :

a. Pemulusan Eksponensial Tunggal : - $S'_t = \alpha X_t + (1 - a)\, S'_{t-1}$

b. Pemulusan Eksponensial Ganda : - $S''_t = \alpha S'_t + (1 - a)\ S''_{t-1}$

c. Pemulusan Trend : - $\alpha_t = 2S'_t - S''_t$ - $\boldsymbol{b_t = \alpha\ 1 - a\ (S'_t - S''_t)}$

d. Ramalan : - $F_{t+m} = \alpha_t + b_t(m)$

Keterangan :

S'_t = Nilai pemulusan tunggal S''_t

= Nilai pemulusan ganda

Xt = Data sebenarnya pada waktu ke t

m = Jumlah periode bulan yang akan diramalkan

2.3 Metode Mengukur *Error* dalam *Forecasting*

(Hanke & Wichern, 2005) mengatakan bahwa di dalam teknik *forecasting* yang menggunakan data kuantitatif sering terdapat data berupa runtun waktu tertentu. Yang dimana biasa terdapat *error* / kesalahan yang dilakukan oleh teknik *forecasting*. Oleh sebab itu dibutuhkan metode untuk mengukur seberapa besar *error* / kesalahan yang dapat dihasilkan oleh metode – metode *forecasting* untuk dipertimbangkan kembali sebelum membuat keputusan.

2.3.1 Mean Absolute Deviation (MAD)

Mean absolute deviation mengukur akurasi dari *forecast* dengan membuat sama rata dari besarnya kesalahan perkiraan yang dimana setiap *forecasting* memiliki nilai *absolut* untuk setiap *error*nya. Rumus yang dipakai untuk menghitung MAD adalah :

$$MAD = \frac{1}{n}\sum_{t=1}^{n} |Y_t - \hat{Y}_t|$$

Gambar 2.2 Formula Eror Mad

Keterangan:

Y_t = Nilai actual pada periode t

$\hat{Y}_t$ = Nilai forecast pada periode t

2.3.2 Mean *Squared Deviation* (MSD)

(Minitab Inc, 2016) mengatakan Mean *squared deviation* (MSD) biasanya dipakai untuk mengukur akurasi dari nilai *time series* yang mau dihitung. Dimana MSD biasanya memiliki efek lebih besar dibandingkan MAD. Rumus yang dipakai untuk menghitung MSD adalah:

$$MSD = \frac{\sum_{t=1}^{n} |Y_t - \hat{Y}_t|^2}{n}$$

Gambar 2.3 Formula Eror Msd

Keterangan:

Y_t = Nilai actual pada periode t

$\hat{Y}_t$ = Nilai forecast pada periode t

2.3.3 Mean Absolute Percentage Error (MAPE)

Mean absolute percentage error dihitung dengan cara mencari *error*/kesalahan absolut di setiap periode yang dimana dibagi dengan nilai *observasi* yang aktual pada periode itu, dan dibuat rata – rata dari *absolute percentage error* tersebut. Rumus yang dipakai untuk menghitung MAPE adalah :

$$MAPE = \frac{1}{n}\sum_{t=1}^{n}\frac{|Y_t - \hat{Y}_t|}{Y_t}$$

Gambar 2.4 Formula Eror Mape

Keterangan:

Y_t = Nilai actual pada periode t

$\hat{Y}_t$ = Nilai forecast pada periode t

Nilai MAPE dapat diinterpretasikan atau ditafsirkan ke dalam 4 kategori yaitu :

1. < 10 % = Sangat Baik
2. 10 % - 20 % = Baik
3. 20 % - 50 % = Cukup Baik
4. > 50 % = Tidak Baik

Semakin kecil nilai MAPE maka semakin kecil kesalahan hasil dari peramalan, sebaliknya jika semakin besar nilai MAPE maka semakin besar pula kesalahan hasil peramalannya. Hasil peramalan mempunyai kemampuan yang sangat baik jika nilai MAPE < 10 %.

2.4 Pemrograman Aplikasi

2.4.1 PHP(*Hypertext Preprocessor*)

Menurut Priyanto Hidayatullah & Jauhari Khairul Kawistara (2017), PHP (*Hypertext PreProcessor*) adalah bahasa komputer/bahasa pemrograman / koding / script yang digunakan untuk mengolah data dari server untuk ditampilkan di website. PHP digunakan untuk membuat *website* dinamis. Dalam penggunaan murninya, kode-kode PHP disisipkan diantara kode HTML. Secara *default*, dokumen PHP memiliki ekstensi .php. Saat *server web* menemukan web menemukan file dengan jenis ini, file tersebut secara otomatis dikirim untuk diproses oleh prosesor PHP.

2.4.2 Javascript

JavaScript merupakan bahasa scripting yang pada awalnya dikembangkan oleh *Netscape*. Dengan menggunakan *javascript*, dapat dibuat tampilan web yang lebih interaktif lagi. Agar user dapat menjalankan javascript, alat yang dibutuhkan hanyalah browser yang kemampuan javascript-nya telah diaktifkan. Penulisan javascript dapat dipadukan pada halaman HTML, dan dari sini dapat dilihat bahwa javascript merupakan bahasa pemrograman yang berjalan di sisi client. Karena proses dari javascript tergantung proses dari *browser* yang digunakan oleh *user*. *JavaScript* pada penelitian ini digunakan untuk perancangan kuis.

2.4.3 Database

Menurut Mukhamad Masrur (2016), *Database* adalah sekumpulan *file* data yang satu sama lainnya saling berhubungan yang diorganisasi sedemikian rupa

sehingga memudahkan untuk mendapatkan dan memproses data tersebut. Lingkungan sistem database menekankan pada data yang tidak tergantung (*independent*) pada aplikasi yang akan menggunakan data tersebut. Penggunaan database pada komputer dilakukan dengan menggunakan table. Pada tabel-tabel tersebut masih dikelompokkan lagi menjadi beberapa bagian untuk membedakan data yang satu dengan data yang lain. Pada sebuah tabel database harus memiliki setau kategori data yang digunakan sebagai kunci untuk membedakan data-data yang ada didalam satu tabel. Data kunci tersebut tidak boleh sama antara satu data dengan data lainnya. Data kunci sering disebut dengan *Primary Key*.

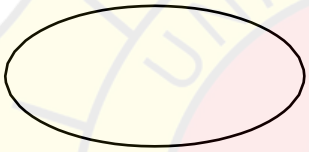
2.5 *Unified Modeling Language* (UML)

"*Unified Modeling Language*" bukanlah suatu proses melainkan bahasa pemodelan secara grafis untuk menspesifikasikan, memvisualisasikan, membangun, dan mendokumentasikan seluruh artifak sistem perangkat lunak. Penggunaan model ini bertujuan untuk mengidentifikasikan bagian-bagian yang termasuk dalam lingkup sistem dibahas dan bagaimana hubungan antara sistem suatu dengan subsistem maupun sistem lain di luarnya.

" *Unified Modeling Language* " adalah sebuah bahasa yang berdasarkan *grafik* atau gambar untuk memvisualisasi, menspesifikasikan dari sebuah sistem pengembangan *software* berbasis *object oriented*." Dari Pengertian diatas penulis menyimpulkan bahwa *Unified Modeling Language* merupakan bahasa pemodelan yang berbentuk grafis yang digunakan untuk memvisualisasi, menspesifikasikan suatu sistem perangkat lunak (Yuhanar, 2018).

2.5.1 Use Case Diagram

Diagram use case merupakan pemodelan untuk kelakuan (*behavior*) sistem informasi yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. Secara kasar, *use case* digunakan untuk mengetahui fungsi apa saja yang ada di dalam suatu sistem dan siapa saja yang berhak menggunakan fungsi-fungsi tersebut." (Yunahar, 2018). Berikut adalah simbol-simbol yang ada pada diagram use case:

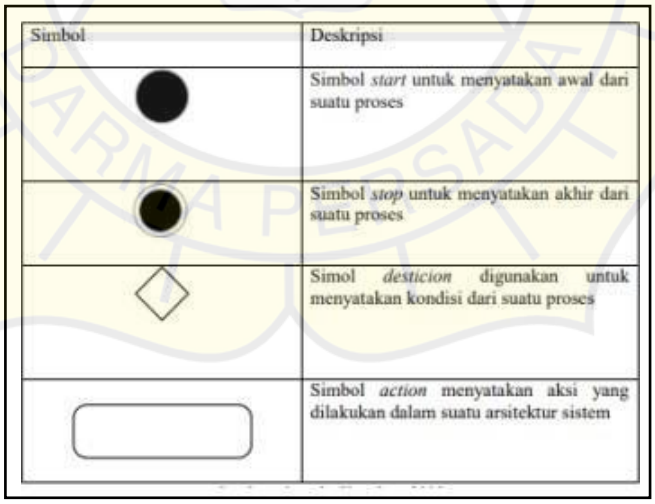
SIMBOL	DESKRIPSI
Use Case 	Fungsional yang disediakan sistem sebagai unit-unit yang saling bertukar antar unit atau aktor biasaanya dinyatakan dengan menggunakan kata kerja diawal frase nama usecase
Aktor 	Orang,proses atau sistem lain yang berinteraksi dengan sistem informasi yang akan di buat diluar sistem informasi yang akan dibuat itu sendiri,jadi walaupun simbol dari aktor adalah gambar orang biasanya dinyatakan menggunakan kata benda diawal frase nama aktor
Asosiasi	komunikasi antara aktor dan usecase yang berpatisipasi pada usecase atau use case memiliki interaksi dengan aktor
Extensi	Relasi Use case tambahkan kesebuah usecase dimana usecase yang ditambahkan dapat berdiri sendiri walau tampa usecase tambahan

Generalisasi 	Hubungan generalisasi dan sepealisasi(Umum-Khusus) anatar dua buah usecase dimana fungsi yang satu adalah fungsi yang lebih umum dari yang lainnya
Include 	Relasi usecase tambahkan ke sebuah usecase dimana usecase yang ditambahkan memerlukan usecase ini untuk menjalankan fungsinya atau sebagai syarat diajalankan usecase ini

Gambar 2.5 Usecase Diagaram

2.5.2 Activity Diagam

Activity Diagram menggambarkan *work flow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis. Yang perlu diperhatikan disini adalah bahwa diagram aktivitas menggambarkan aktivitas sistem bukan apa yang dilakukan aktor, jadi aktivitas dapat dilakukan oleh sistem" (Yunahar, 2018).

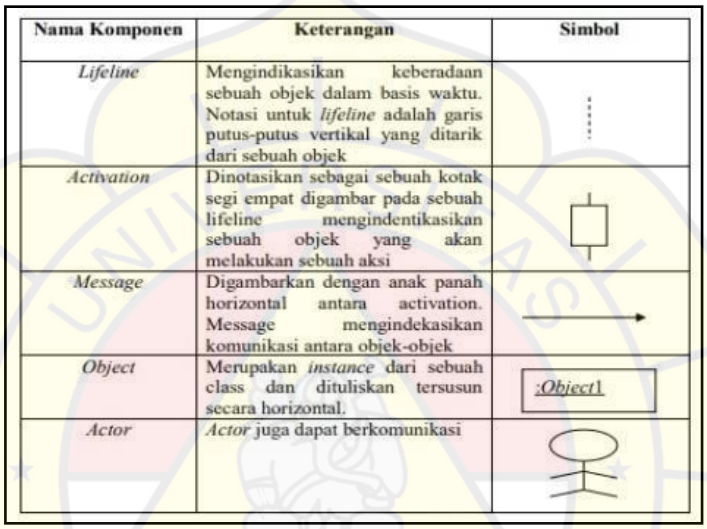


Simbol	Deskripsi
	Simbol *start* untuk menyatakan awal dari suatu proses
	Simbol *stop* untuk menyatakan akhir dari suatu proses
	Simol *desticion* digunakan untuk menyatakan kondisi dari suatu proses
	Simbol *action* menyatakan aksi yang dilakukan dalam suatu arsitektur sistem

Gambar 2.6 Activity Diagram

2.5.3 Sequence Diagram

"*Sequence Diagram* adalah *tool* yang sangat populer dalam pengembangan sistem informasi secara *object-oriented* untuk menampilkan interaksi antar objek. Berdasarkan definisi tersebut, dapat disimpulkan bahwa *Sequence Diagram* adalah *tool* yang digunakan dalam pengembangan *system*" (Yunahar, 2018).



Nama Komponen	Keterangan	Simbol
Lifeline	Mengindikasikan keberadaan sebuah objek dalam basis waktu. Notasi untuk *lifeline* adalah garis putus-putus vertikal yang ditarik dari sebuah objek	
Activation	Dinotasikan sebagai sebuah kotak segi empat digambar pada sebuah lifeline mengindentikasikan sebuah objek yang akan melakukan sebuah aksi	
Message	Digambarkan dengan anak panah horizontal antara activation. Message mengindekasikan komunikasi antara objek-objek	
Object	Merupakan *instance* dari sebuah class dan dituliskan tersusun secara horizontal.	:Object1
Actor	*Actor* juga dapat berkomunikasi	

Gambar 2.7 Sequence Diagram

2.5.4 Deployment Diagram

Menurut Sukamto dan Shalahudin (2015) mengemukakan bahwa "*Deployment Diagram* menunjukan konfigurasi komponen dalam proses eksekusi aplikasi". Deployment Diagram dapat digunakan untuk menggambarkan hal-hal sebagai berikut:

1. Sistem tambahan (*embedded system*) yang menggambarkan rancangan *device*, *node* dan *hardware*.

2. Sistem *client* atau *server*.

3. Sistem terdistribusi murni.

4. Rekayasa ulang aplikasi.

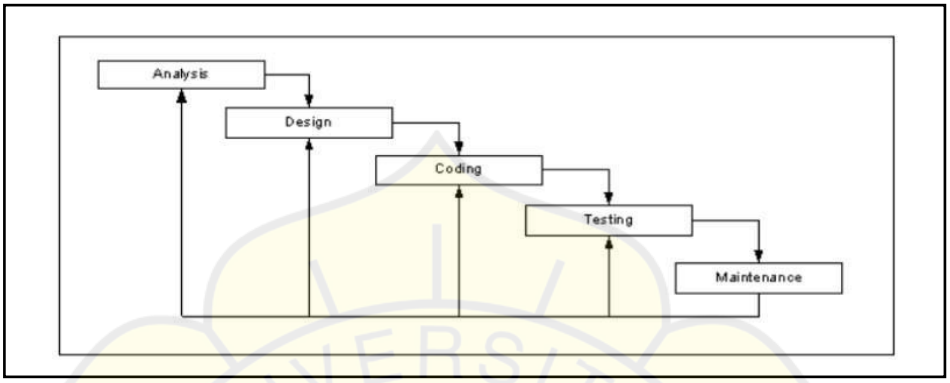
Menurut Hendini (2016) *Deployment Diagram* digunakan untuk menggambarkan detail bagaimana komponen disusun di infrastruktur sistem. Dapat disimpulkan bahwa *Deployment Diagram* merupakan diagram yang digunakan untuk menggambarkan konfigurasi pada sistem.

Gambar 2.8 Deployment Diagram

2.6 Metodelogi Pengembangan Sistem

Menurut Pressman, R.S. (2015), Metodologi *waterfall* adalah suatu proses pengembangan perangkat lunak berurutan, di mana kemajuan dipandang sebagai terus mengalir ke bawah (seperti air terjun) melewati fase perencanaan, pemodelan, implementasi (konstruksi), dan pengujian.

Adapun penjelasan urutan dari tahapan-tahapan yang dimiliki metodologi *waterfall* adalah sebagai berikut:



Gambar 2.9 Metodologi Waterfall Menurut Pressman, Sommerville

2.6.1 Analisis

Langkah ini merupakan analisa terhadap kebutuhan sistem. Pengumpulan data dalam tahap ini bisa melakukan sebuah penelitian, wawancara atau study literature. Seseorang sistem analisis akan menggali informasi sebanyak-banyaknya dari user sehingga akan tercipta sebuah sistem komputer yang bisa melakukan tugas-tugas yang diinginkan oleh user tersebut.

2.6.2 Desain

Proses design akan menterjemahkan syarat kebutuhan kesebuah perancangan perangkat lunak yang dapat diperkirakan sebelum dibuat koding. Proses ini berfokus pada struktur data, arsitektur perangkat lunak, *representasi interface*, dan detail (algoritma) prosedural. Tahapan ini akan menghasilkan dokumen yang disebut software requirement. Dokumen inilah yang akan digunakan programmer untuk melakukan aktivitas pembuatan sistemnya.

2.6.3 Pengkodean

Tahapan ini merupakan proses penerjemahan bentuk desain menjadi kode atau bentuk/bahasa yang dapat dibaca oleh mesin. Setelah pengkodean selesai, dilakukan pengujian terhadap sistem dan juga kode yang sudah dibuat. Tujuannya untuk menemukan kesalahan yang mungkin terjadi untuk nantinya diperbaiki.

2.6.4 Pengujian Program (*Testing*)

Tahapan ini bisa dikatakan final dalam pembuatan sebuah sistem. Setelah melakukan analisa, design dan pengkodean maka sistem yang sudah jadikan digunakan oleh user.

2.6.5 Pemeliharaan / Perawatan (*Maintenance*)

Perangkat lunak yang sudah disampaikan kepada pelanggan pasti akan mengalami perubahan. Perubahan tersebut bisa karena mengalami kesalahan karena perangkat lunak harus menyesuaikan dengan lingkungan baru, atau karena pelanggan membutuhkan perkembangan fungsional.

TEKNOLOGI INFORMASI

UNIVERSITAS DARMA PERSADA