

BAB II

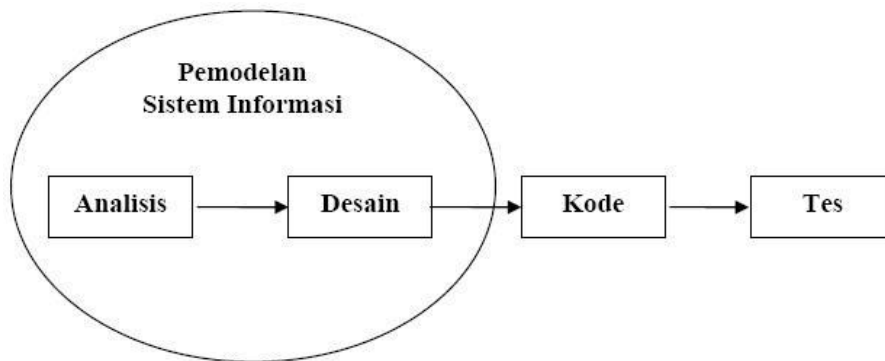
LANDASAN TEORI

2.1. Tinjauan Pustaka

A. Konsep Dasar Sistem Informasi

Menurut Rosa dan Shalahuddin (2013:28) “model *SDLC* air terjun (*waterfall*) sering juga disebut model sekuensial linier (*sequential linear*) atau alur hidup klasik (*classic life cycle*)”.

Model air terjun menyediakan pendekatan alur hidup perangkat lunak secara sekuensial atau terurut dimulai dari analisis, desain, pengkodean, pengujian, dan tahap pendukung (*support*). Berikut adalah gambar model air terjun :



Sumber: Rosa dan Shalahuddin (2013:28)

Gambar II.1. Waterfall Rosa

1. Analisis kebutuhan perangkat lunak

Proses pengumpulan kebutuhan dilakukan secara intensif untuk menspesifikasikan kebutuhan perangkat lunak seperti apa yang dibutuhkan oleh

user. Spesifikasi kebutuhan perangkat lunak pada tahap ini perlu untuk didokumentasikan.

2. Desain

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

3. 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.

4. Pengujian

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

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

Tidak menutup kemungkinan sebuah perangkat lunak mengalami perubahan ketika sudah dikirimkan ke *user*. Perubahan bisa terjadi karena adanya kesalahan yang muncul dan tidak terdeteksi saat pengujian atau saat perangkat lunak harus beradaptasi dengan lingkungan baru. Tahap pendukung atau

pemeliharaan dapat mengulangi proses pengembangan mulai dari analisis spesifikasi untuk perubahan perangkat lunak yang sudah ada, tapi tidak untuk membuat perangkat lunak baru.

B. Konsep Dasar Pemrograman

Menurut Al-Al-Bahra (2010:259) “Pemrograman terstruktur adalah cara pengorganisasian dan pengkodean program-program yang kompleks dan membuat program tersebut lebih mudah dipahami dan dimodifikasi”.

Langkah-langkah dalam pengembangan sistem adalah sebagai berikut :

1. Definisikan Masalah

Pemahaman permasalahan merupakan suatu hal yang sangat penting bagi si pemogram. Si pemogram perlu memahami permasalahan yang dihadapi dan yang akan diselesaikan oleh pemesan program, agar hasil pendefinisian masalah tidak menyimpang dari masalah yang sedang dihadapi.

2. Merancang *Outline* Pemecahan Masalah

Dalam merancang *outline* pemecahan masalah pemogram perlu membuat rincian proses secara rinci (meliputi proses apa saja yang terjadi pada menu utama, lalu masing-masing submenu dalam menu utama tersebut akan mengerjakan berapa proses, setiap proses yang terjadi berinteraksi dengan *file* apa saja, dan berapa *file* yang terlibat dalam proses tersebut).

3. Pengkodean/Menulis Program

Tujuan pengkodean adalah terjadinya efisiensi terhadap memori yang akan digunakan, efisiensi perintah dalam setiap modul program, serta efisiensi

terhadap penggunaan fasilitas *input* dan *output* yang akan berpengaruh langsung terhadap prosesor ataupun pengguna.

4. Eksekusi Program

Dalam mengeksekusi program maka perlu memperhatikan beberapa factor seperti jenis bahasa pemrograman yang dipilih apakah sanggup mengangkut data yang akan diproses dan spesifikasi perangkat keras yang digunakan agar tidak terjadi kesalahan.

5. Dokumentasi Dan Pemeliharaan Program

Pemeliharaan program berfungsi untuk menjabarkan aktivitas dari hasil analisis terhadap sistem dan dilakukan setelah program diimplementasikan dan telah digunakan beberapa saat oleh pemakai. Pemeliharaan mencakup format tampilan yang disesuaikan, fungsi-fungsi yang tidak sesuai oleh pengguna dan adaptasi dengan spesifikasi prosesor yang baru.

Komponen Dalam Pemrograman Terstruktur adalah sebagai berikut :

a. Pemrograman Top-Down dan *Button-Up*

Teknik *Top-Down* sebagai suatu masalah yang kompleks dibagi-bagi kedalam beberapa kelompok masalah yang lebih kecil. Sedangkan Teknik *Botton-Up* dilakukan dengan menggabungkan prosedur-prosedur yang ada menjadi satu kesatuan program guna menyelesaikan masalah.

b. Pemrograman Modular

Teknik pemrograman modular yang digunakan untuk mengimplementasikan langkah-langkah pemecahan masalah pada kelompok masalah. Modul

Program sebagai sekumpulan instruksi yang memiliki operasi-operasi dan data yang didefinisikan.

c. Struktur Kontrol

Teknik pemograman yang menggunakan tiga struktur yaitu dengan struktur urut, struktur seleksi dan struktur pengulangan dalam menyelesaikan masalah. Untuk mendukung dalam penyusunan skripsi ini, penulis menggunakan *software* bahasa pemrograman :

C. Tool (Peralatan Pendukung)

1. Adobe Photoshop CS3

Adobe Photoshop CS3 dalam Madcoms (2008:49) adalah “salah satu program grafis yang dapat digunakan untuk mengolah gambar dan membuat desain dalam pembuatan situs web”.

Adobe Photoshop CS3 juga merupakan suatu program pengolahan image atau gambar Bitmap. Image atau gambar Bitmap sering disebut dengan Raster.

Photoshop dalam madcoms (2012:20) “merupakan program penyuntingan gambar yang paling populer hingga saat ini”.

Kemampuan serta fasilitasnya yang lengkap membuatnya diminati oleh para seniman, *professional*, maupun pemula yang membutuhkan sebuah program gambar yang lengkap namun mudah dalam penggunaannya.

Photoshop saat ini digunakan untuk membuat gambar-gambar untuk keperluan seperti iklan, brosur, poster, serta sebagai macam output cetak lainnya bagi para penggemar fotografi, *photoshop* juga dapat digunakan untuk

meretouch foto yang sudah rusak, sehingga memodifikasi foto sehingga menjadi lebih baik.

2. **PHP**

Menurut Peranginangin (2008:2) mengatakan bahwa “singkatan dari (*Hypertext Preprocessor*) yang digunakan sebagai bahasa *scriptserver-side* dalam pengembangan *web* yang disisipkan pada dokumen HTML”.

3. **Pengenalan MySQL**

Menurut Nugroho (2008:91) memberikan batasan bahwa “*Mysql* (*My Structured Query Language*) atau yang biasa dibaca ***mai-se-kuel*** adalah sebuah program pembuat dan pengelola database atau yang sering disebut dengan DBMS (*Database Management System*), sifat dari DBMS ini adalah *Open Source*”.

4. **Pengenalan HTML**

Menurut Junaedi (2010:19) mengatakan bahwa “suatu format data yang digunakan untuk membuat dokumen *hypertext* yang dapat dieksekusi dari satu *platform* komputer ke *platform* komputer lainnya tanpa perlu melakukan suatu perubahan apapun dengan suatu alat tertentu”.

5. **Pengenalan Web Browser**

Menurut Sutarman (2007:261) mengatakan bahwa “suatu program yang dirancang untuk mengambil informasi dari suatu *server* komputer pada jaringan internet”.

C. UML (*Unified Modelling Language*)

Menurut Fowler (2010:1) “*Unified Modelling Language* (UML) adalah keluarga notasi grafis yang didukung oleh meta-model tunggal, yang membantu pendeskripsian dan desain sistem perangkat lunak, khususnya sistem yang dibangun menggunakan pemrograman berorientasi objek (OOP)”.

a. *Use case diagram*

Use case adalah teknik untuk merekam persyaratan fungsional sebuah sistem. *Use case* mendeskripsikan interaksi tipikal antara para pengguna sistem dengan sistem itu sendiri, dengan memberi sebuah narasi tentang bagaimana sistem tersebut digunakan.

b. *Activity diagram*

Activity diagram adalah teknik untuk menggambarkan logika prosedural, proses bisnis, dan jalur kerja. Dalam beberapa hal, diagram ini memainkan peran mirip sebuah diagram alir, tetapi perbedaan prinsip antara diagram ini dan notasi diagram alir adalah diagram ini mendukung *behavior paralel*.

Menurut Munawar (2010:109) *Activity Diagram* adalah teknik untuk mendeskripsikan logika procedural, proses bisnis dan aliran kerja dalam banyak kasus. *Activity Diagram* mempunyai peran seperti halnya *Flowchart*, akan tetapi perbedaannya dengan *Flowchart* adalah *Activity Diagram* bisa mendukung perilaku paralel sedangkan *Flowchart* tidak bisa.

c. *Class diagram*

Class diagram mendeskripsikan jenis-jenis objek dalam sistem dan berbagai macam hubungan statis yang terdapat diantara mereka. *Class diagram* juga menunjukkan properti dan operasi sebuah *class* dan batasan-batasan yang terdapat dalam hubungan-hubungan objek tersebut. UML menggunakan

istilah fitur sebagai istilah umum yang meliputi properti dan operasi sebuah *class*.

d. *Sequence diagram*

Menurut Fowler (2010:81) “*Interaction diagram* menunjukkan bagaimana kelompok-kelompok objek saling berkolaborasi dalam beberapa *behavior*, UML memiliki beberapa bentuk *interaction diagram* dan yang paling umum digunakan adalah *sequence diagram*”.

Sebuah *sequence diagram*, secara khusus menjabarkan *behavior* sebuah skenario tunggal. Diagram tersebut menunjukkan sejumlah objek contoh dan pesan-pesan yang melewati objek-objek ini di dalam *use case*.

e. *Component diagram*

Menurut Fowler (2010:189) “*Component Diagram* tidak seperti ikon, komponen tidak menggunakan notasi yang asing dengan kita. Komponen terhubung melalui antarmuka yang digunakan dan dibutuhkan, sering menggunakan notasi bola dan soket seperti halnya *class diagram*”.

Anda dapat juga memecah komponen dengan menggunakan *composite structure diagram*.

f. *Deployment diagram*

Menurut Fowler (2010:137) “*Deployment diagram* menunjukkan susunan fisik sebuah sistem, menunjukkan bagian perangkat lunak mana yang berjalan pada perangkat keras mana”.

D. ERD (*Entity Relationship Diagram*)

Menurut Al-Bahra (2010:142) “ERD adalah suatu model jaringan yang menggunakan susunan data yang di simpan dalam sistem secara abstrak”.

Entity Relation Diagram (ERD) digunakan untuk menggambarkan hubungan antara *datastore* yang ada di dalam diagram hubungan data antara lain:

a. *Entity* (entitas)

Entity di gambarkan dengan sebuah bentuk persegi panjang. *Entity* adalah sesuatu apa saja yang di dalam sistem, nyata maupun abstrak dimana data tersimpan atau dimana terdapat data. Entitas diberi nama dengan kata benda dan dapat di kelompokkan dalam empat jenis nama, yaitu orang, benda, lokasi, kejadian (terdapat unsure waktu didalamnya).

b. *Relationship* (hubungan)

Relationship dapat digambarkan dengan sebuah bentuk belah ketupat. *Relationship* adalah hubungan alamiah yang terjadi antara *entitas*. Pada umumnya *relationship* diberi nama dengan kata kerja, sehingga memudahkan untuk melakukan pembacaan relasinya. Penggambaran hubungan yang terjadi adalah sebuah bentuk belah ketupat dihubungkan dengan dua bentuk persegi panjang.

c. *Atribut*

Atribut adalah sifat atau karakteristik dari tiap *entitas* maupun tiap *relationship*. Menjelaskan apa yang sebenarnya yang dimaksud entitas maupun *relationship*, sehingga atribut adalah *element* dari setiap *entitas* dan *relationship*.

d. *Cardinality* (kardinalitas)

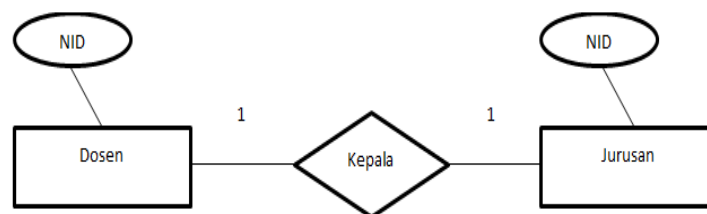
Kardinalitas Relasi menunjukan jumlah *maksimum tupelo* yang dapat berelasi dengan *entitas* pada entitas yang lain.

Kardinalitas relasi merujuk kepada hubungan *maksimum* yang terjadi dari *entitas* yang satu ke *entitas* yang lain dan begitu juga sebaliknya.

Terdapat 3 macam kardinalitas relasi, yaitu:

1. *Satu ke satu (one to one 1 : 1)*

Tingkat hubungan satu ke satu, dinyatakan dengan suatu kejadian pada *entitas* pertama, hanya mempunyai satu hubungan dengan satu kejadian pada *entitas* yang kedua sebaliknya.



Menurut Al-Bahra (2010:160)

Gambar II.2 Diagram Kardinalitas One To One

2. *Satu banyak (one to many atau 1 : M)*

Tingkat hubungan satu banyak adalah sama dengan banyak ke satu.

Untuk satu kejadian pada entitas yang pertama dapat mempunyai banyak hubungan dengan kejadian pada *entitas* yang kedua. Sebaliknya satu kejadian pada *entitas* kedua hanya dapat mempunyai satu hubungan dengan satu kejadian pada *entitas* yang pertama.

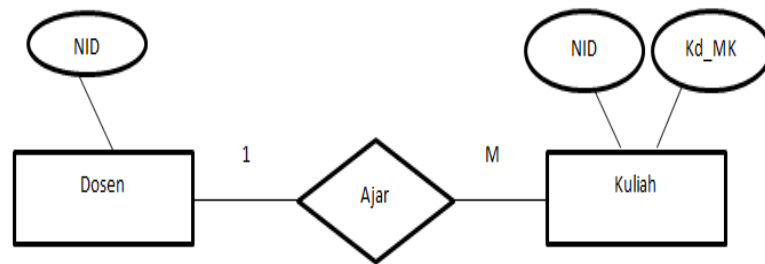
a. *One to Many* (satu ke banyak)

Satu *tupel* pada *entitas* A dapat berhubungan dengan banyak *tupelo* pada entitas B, tetapi tidak sebaliknya, dimana setiap *tupel* pada entitas B,

berhubungan dengan paling banyak satu tupel pada entitas A.

b. *Many to One* (banyak ke satu)

Setiap *tupelo* pada *entitas* A dapat berhubungan dengan paling banyak satu *tupelo* pada entitas B, tetapi tidak sebaliknya, dimana setiap *tupelo* pada entitas A berhubungan dengan paling banyak satu tupel pada *entitas* B.

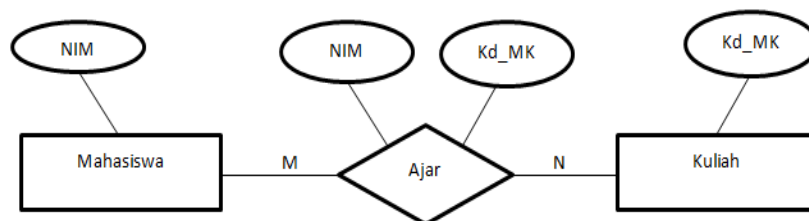


Menurut Al-Bahra (2010:165)

Gambar II.3 Diagram Kardinalitas One To Many

3. Banyak ke banyak (*Many to Many* atau $M : M$)

Tingkat hubungan banyak ke banyak terjadi jika tiap kejadian pada sebuah *entitas* akan mempunyai banyak hubungan dengan kejadian pada *entitas* lainnya. Baim dilihat dari si *entitas* yang pertama, maupun dilihat dari sisi yang kedua.



Menurut Al-Bahra (2010:169)

Gambar II.4 Diagram Kardinalitas Many To Many

2.2. Penelitian Terkait

Menurut Widarda, Syam dan Hakim, Nur Fitro(2014:67) usaha yang bersaing dalam industri *coffee kiosk*. Pada bulan Juli (2012) Coffee Toffee mempunyai pelanggan sebanyak 1500 pelanggan dengan penjualan bersih tiap bulan kira-kira sebesar Rp 15.000.000,-. Rata-rata *order* penjualan yang masuk tiap bulan 2023 *order* penjualan. Kenaikan penjualan untuk bulan juli 29%, tingkat penurunan penjualan bulan Agustus adalah 27% dan untuk bulan September sebesar 44%. Penurunan kenaikan tingkat penjualan ini diperkirakan adanya pelaku bisnis yang serupa dengan Coffee Toffee sehingga pelayanan dan kesan yang berbeda sangat diperlukan dalam persaingan bisnis tersebut. Salah satu cara yang dapat dilakukan pihak Coffee Toffee yaitu berusaha memberikan produk dan layanan yang terbaik agar dapat mempertahankan konsumen, mendapatkan konsumen baru, dan menciptakan loyalitas konsumen.

Menurut Ibrahim, Ali (2011:321) Salah satu aspek keunggulan yang harus dibangun adalah pengelolaan sistem informasi yang dapat menjangkau masa lalu, masa kini dan masa depan, tersedia dalam segala bentuk *output* komputer dan digunakan oleh para manajer maupun non manajer dalam memecahkan masalah. Hal lain yang sangat penting untuk dipertimbangkan adalah adanya sistem basis data. Sistem penjualan tiket pesawat berbasis web ini pengembangannya difokuskan pada permasalahan web database dan web desain yang mempunyai tujuan untuk memberikan informasi tentang pemesanan tiket secara online dengan menggunakan media web yang dapat mempermudah para konsumen atau pengguna jasa penerbangan, dimana para konsumen tidak perlu untuk datang kepihak LionAir hanya untuk melakukan pemesanan, tapi dengan adanya web ini dapat langsung melakukan pemesanan tiket secara langsung lewat web yang tersedia