BAB II

LANDASAN TEORI

2.1. Konsep Dasar Web

Dalam merancang sebuah website terdapat beberapa hal yang perlu diketahui sebelum membangun *website* tersebut, diantaranya adalah :

A. *Website*

Web dibuat dengan suatu bahasa pengkodean HTML, agar dapat interaktif maka seorang *web development* membuat suatu pemrograman agar dapat berinteraksi antara pengunjung dan situs tersebut, ada banyak bahasa yang dapat digunakan seperti ASP, PHP, *Javascript*, Css, XML, CMS dan lain-lain.

> Menurut Hidayat pada Hikmah (2010:1) :
> "*Website* atau situs dapat diartikan sebagai kumpulan halaman yang digunakan untuk menampilkan informasi teks, gambar diam atau gerak, animasi, suara, dan atau gabungan dari semuanya, baik yang bersifat statis maupun dinamis yang membentuk suatu rangkaian bangunan yang saling terkait, yang masing-masing dihubungkan dengan jaringan-jaringan halaman".

Hidayat (2010:1) Dunia internet berkembang dengan sangat pesat seakan-akan telah menjadi bagian hidup masyarakat moderen saat ini. Betapa tidak, karena internet secara lengkap menyediakan kebutuhan akan informasi, berita, serta ilmu pengetahuan. Dengan internet seolah-olah tidak ada lagi batasan antar ruang dan waktu dalam berkomunikasi dengan berbagai orang diberbagai belahan dunia. Sebagai konsumen dari teknologi *web* tentunya mengharapkan tampilan layar yang mengasyikan serta mudah dipakai dan dimanfaatkan. Misalnya saja, jika kita hendak membeli sesuatu, kita tinggal

mengakses *e-commerece*, kemudian melakukan transaksi jual beli secara *online* dan barang yang dibeli akan sampai di rumah kita. Begitu juga halnya, kalau ingin kuliah, kita tinggal mendaftar pada *website-weebsite* yang menyediakan jasa layanan *e-learning*, proses perkuliahan dapat dilakukan secara *online* walau dibatasi oleh jarak. Tidak itu saja, sampai dengan pemesanan tiket pesawat, pemesanan makanan, transaksi perbankan, *e-goverment*, dan lain sebagainya, semuanya dapat dilayani oleh internet melalui media yang disebut *website*. Pada dasarnya *web* merupakan suatu kumpulan *hyperlink* yang menuju dari alamat satu ke alamat lainnya dengan bahasa HTML (*HyperText Makrup Languange*).

Penemu situs *web* adalah Timothy Jhon Berenrs-Lee, sedangkan situs *web* yang tersambung dengan jaringan pertama kali muncul pada tahun 1991. Maksud dari Timothy ketika merancang situs *web* adalah untuk memudahkan tukar menukar dan memperbarui informasi pada tanggal 30 April 1993.

Situs *web* biasanya ditempatkan pada *server web*. Sebuah *server web* umumnya telah dilengkapi dengan perangkat-perangkat lunak khusus untuk menangani pengaturan nama ranah, serta menangani layanan atas *protocol* HTTP yang disebut sebagai *server* HTTP seperti *Apache* HTTP *server* atau *internet informatika service* (IIS).

Menurut Hidayat (2010:3) jenis-jenis web berdasarkan sifat atau *style*-nya yaitu:

1. *Website Dinamis*, merupakan sebuah *website* yang menyediakan konten atau isi yang selalu berubah-ubah setiap saat. Bahasa pemrograman

yang digunakan antara lain PHP, ASP, .NET dan memanfaatkan *database Mysql* atau *MS SQL*.

2. *Website Statis*, merupakan *website* yang kontennya sangat jarang berubah. Bahasa pemrograman yang digunakan adalah HTML dan belum memanfaatkan *database*.

Berdasarkan pada fungsinya, *website* terbagi atas:

1. Personal *website*, *website* yang berisi informasi pribadi.
2. *Commercial website*, *website* yang dimiliki oleh sebuah perusahaan yang bersifat bisnis.
3. *Government website*, *website* yang dimiliki oleh instansi pemerintah, pendidikan, yang bertujuan memberikan pelayanan kepada pengguna.
4. Non-profit *Organization website*, dimiliki oleh organisasi yang bersifat non-profit atau tidak bersifat bisnis.

Website mempunyai fungsi yang bermacam-macam tergantung dari tujuan dan jenis *website* yang dibangun, tetapi secara garis besar fungsi dari *website* (Abrams, 2008:168) adalah:

1. Informasional

   Salah satu kegunaan terbaik dari sebuah *website* adalah untuk memberikan informasi detail tentang perusahaan anda kepada para calon konsumen dan pegawai yang mendengar kabar tentang anda secara *offline* (bukan lewat internet).

2. Relasional

   Sebuah *website* bisa menjadi secara cukup baik dalam menjalin hubungan yang lebih erat dengan konsumen anda saat ini. Anda bisa

menampilkan informasi-informasi khusus dan detail tentang topik-topik yang berkaitan dengan perusahaan anda.

B. Bahasa Pemrograman

Menurut Kusrini (2007:171) “Bahasa pemrograman adalah perintah-perintah yang dimengerti oleh komputer untuk melakukan tugas-tugas tertentu”. Berikut terdapat beberapa bahasa pemrograman.

1. HTML (*Hyper Text Makrup Lanuage*)

   HTML merupakan bahasa pemrograman *web* yang memiliki sintak atau aturan tertentu dalam menuliskan *script* atau kode-kode HTML.

   Menurut Anhar (1010:40) “HTML (*Hyper Text Makrup Language*) adalah sekumpulan simbol-simbol atau tag-tag yang dituliskan dalam file yang digunakan untuk menampilkan halaman pada *web browser*. Tag-tag HTML selalu diawali dengan <x> dan diakhiri </x> dimana x tag HTML itu seperti b, i, u, dll”.

   Menulis Kode HTML. Program yang digunakan untuk membuat *document* HTML mengunakan HTML *Editor*. Ada banyak HTML *editor* yang bisa digunakan, diantaranya : *Notepad*, *Ms. FrontPage*, dan *Dreamweaver*.

   Setiap dokumen HTML harus diawali dan ditutup dengan tag HTML <HTML></HTML>. Tag HTML memberi tahu browser bahwa yang ada di dalam kedua tag tersebut adalah *document* HTML. Bagian *Header* dari *document* HTML didapat oleh *document* HTML. Bagian *Header* dari *document* HTML didapat oleh tag <HEAD></HEAD> di dalam bagian ini biasanya dimuat tag TITLE yang menampilkan judul dari halaman

pada titlenya *browser*. Bagian *Body* dari *documen* HTML didapat oleh tag <BODY></BODY>. *Document body* digunakan untuk menampilkan *text*, *image link*, dan semua yang akan ditampilkan pada halaman *web*. Selanjutnya kita akan membuka file tes.html tersebut dengan *brwoser*.

2. *Personal Home Page* (PHP)

   Bahasa pemrograman php menurut Suparto dalam Hikmah(2010:1) mengemukanan bahwa "php merupakan kependekan dari kata *Hypertext Preprocessor*. PHP tergolong sebagai perangkat lunak *open source* yang diatur dalam aturan *general purpose licences* (GPL)".

   Menuurut Anhar (2010:3) "PHP singkatan dari PHP : *Hypertxt Preprocessor* yaitu bahasa pemrograman *web server-side* yang bersifat *open source*".

   Bahasa pemrograman PHP sangat cocok dikembangan dalam lingkungan *web*, karena PHP bisa diterapkan pada *script* HTML atau sebaliknya. PHP dikhususkan untuk pengembangan *web* dinamis. Maksudnya, PHP mampu menghasilkan *website* yang secara terus-menerus hasilnya bisa berubah-ubah sesuai dengan pola yang diberikan. PHP tergolong juga bahasa pemrograman yang berbasis *server* (*server side scripting*).

   PHP merupakan bahasa standar yang digunakan dalam bahasa *website*. Jika kita lihat dari sejarah, mulanya PHP diciptakan dari ide Rasmus Lerdof yang membuat sebuah *script perl*. *Script* tersebut sebenarnya dimaksudkan untuk digunakan sebagai program untuk dirinya sendiri. Kemudian dikembangkan lagi sehingga menjadi sebuah bahasa yang disebut "*Personal Home Page*". Inilah awal mula munculnya PHP

sampai saat ini. PHP telah diciptakan terutama untuk kegunaan web dan boleh menghubungkan *query database* dan menggunakan *simple task* yang boleh diluruskan dengan tiga atau empat baris kode saja. PHP adalah bahasa *programming* yang baru dibangun sekitar tahun 1994/1995. PHP dapat menukarkan *static website* yang menggunakan HTML ke *dynamic web pages* yang berfungsi secara *automatic* seperti ASP, CGI, dan sebagainya.

PHP sebenarnya program yang berjalan pada *platform* LINUX sehingga membuat program ini menjadi *free ware*. Selanjutnya php mengalami perkembangan yakni dibuat dalam versi *windows*. Alamat *script* php www.php.net. Disana anda akan mendapatkan *script-script* PHP secara gratis mulai dari versi awal sampai versi akhir. Hampir seluruh aplikasi berbasis *web* dapat dibuat dengan PHP ini, namun fungsi PHP yang paling utama adalah untuk menghubungkan *database* dengan *web*. Dengan PHP, membuat aplikasi *web* yang terkoneksi ke *database* menjadi sangat mudah.

Kode PHP mempunyai ciri-ciri khusus (Oktavian, 2010:31), yaitu:

a. Hanya dapat dijalankan menggunakan *web server* misalnya: *Apache*.
b. Kode PHP dapat diletakan dan dijalankan di *web server*.
c. Kode PHP dapat digunakan untuk mengakses *databases*, seperti: MY SQL, PostgreSQL, *Oracle*, dan lain-lain.
d. Merupakan *software* yang bersifat *open source*.
e. Gratis untuk di*download* dan digunakan.

f. Memiliki sistem *multiplatform*, artinya dapat dijalankan menggunakan sistem operasi apapun, seperti *Linux*, *Unix*, *Windows*, dan lain-lain.

3. *Javascript*

Menurut Wahono (2009:97) "*Javascript* adalah bahasa yang berbentuk kumpulan *skrip* yang pada fungsinya berjalan pada suatu dokumen HTML". Bahasa ini adalah bahasa pemrograman untuk memberikan kemampuan tambahan terhadap bahasa HTML dengan mengizinkan pengeksekusian perintah di sisi *user*, yang artinya di sisi *browser* bukan di sisi *server web*.

4. *Java*

Menurut Nofriadi(2015:1) mengemukakan "*java* merupakan satu dari sekian banyak bahasa pemrograman yang dapat dijalankan diberbagai sistem operasi termasuk telepon gengam". Bahasa pemrograman ini pertama kali dibuat oleh James Gosling saat masih bergabung Sun Microsystems. Bahasa pemrograman ini merupakan pengembangan dari bahasa pemrograman C++ karena banyak mengadopsi sintak C dan C++. Saat ini *java* merupakan bahasa pemrograman yang paling populer digunakan, dan secara luas dimanfaatkan dalam pengembangan berbagi jenis perangkat lunak aplikasi atau pun aplikasi basis *web*.

C. Basis Data (*Database*)

Menurut Hutahean (2014:50) Mengemukankan "Basis data merupakan kegiatan sistem program komputer untuk berbagi aplikasi komputer". Dalam basis data dibutuhkan suatu media simpan komputer yang terorganisir sedemikian rupa dan juga pemeliharaan data baik dalam fungsi

manajemen sistam. Pandangan lain bahwa basis data adalah suatu pengetahuan tentang organisasi data, sehingga *database* merupakan salah satu komponen yang penting dalam sistem informasi.

Penerapan *database* dalam sistem informasi disebut dengan sistem basis data (*database system*).

Beberapa rujukan mengatakan bahwa basis data (*Database*) adalalah :

1. Himpunan kelompok data (arsip) yang saling berhubungan yang diorganisasi sedemikian rupa agar kelak dapat dimanfaatkan kembali dengan cepat dan mudah.
2. Kumpulan data yang saling berhubungan yang disimpan secara bersama sedemikian rupa dan tanpa pengulangan (redudansi) yang tidak perlu, untuk memenuhi berbagai kebutuhan.
3. Kumpulan *file*, *table*, arsip yang saling berhubungan yang disimpan dalam media penyimpanan *elektronis*.
4. Basis data (*database*) merupakan kumpulan dari data yang saling berhubungan satu dengan yang lainnya.

   a. Operasi Dasar Basis Data

      Didalam sebuah *disk*, basis data dapat diciptakan dan dapat pula ditiadakan. Kita dapat pula menempatkan beberapa (lebih dari satu) basis data. Sementara dalam sebuah basis data, kita dapat menepatkan satu atau lebih dari satu babel. Pada tabel inilah sesunguhnya data disimpan. Setiap basis data umumnya dibuat untuk mewakili sebuah data yang sepesifik.

Karena itu operasi-operasi dasar yang dapat kita lakukan dengan basis data meliputi:

1. Pembuatan basis data baru (*create database*), yang identik dengan pembuatan lemari arsip yang baru.
2. Penghapusan basis data (*drop database*), yang indentik dengan perusakan lemari arsip (sekaligus beserta isinya, jika ada).
3. Pembuatan tabel baru ke satu basis data (*crate table*), yang identik dengan penambahan map arsip baru ke sebuah lemari arsip yang telah ada.
4. Penghapusan tabel dari sebuah basis data (*drop table*), yang identik dengan perusakan map arsip lama yang ada di sebuah lemari arsip
5. Penambahan atau pegisian data baru ke sebuah label (*query*), yang identik dengan pencarian lembaran arsip dari sebuah map arsip.
6. Pengubahan data dari sebuah tabel (*update*), yang identik dengan perbaikan isi lembaran arsip yang ada di sebuah map arsip.
7. Penghapusan data dari sebuah tabel (*delete*), yang identik dengan penghapusan sebuah lembaran arsip yang ada disebuah mam arsip.

b. Objektif Basis Data

Pemanfaatan basis data dilakukan untuk memenuhi sejumlah tujuan (objektif) seperti berikut ini :

1. Kecepatan dan Kemudahan (*spped*)
2. Efisiensi Rag Penyimpanan (*space*)
3. Kekuatan (*accuracy*)
4. Kesetabilan (*availability*)

5. Kelengkapan (*completeness*)
6. Keamanan (*security*)
7. Kebersamaan Pemakaian (*sharability*)

c. Elemen-elemen dalam *Database*

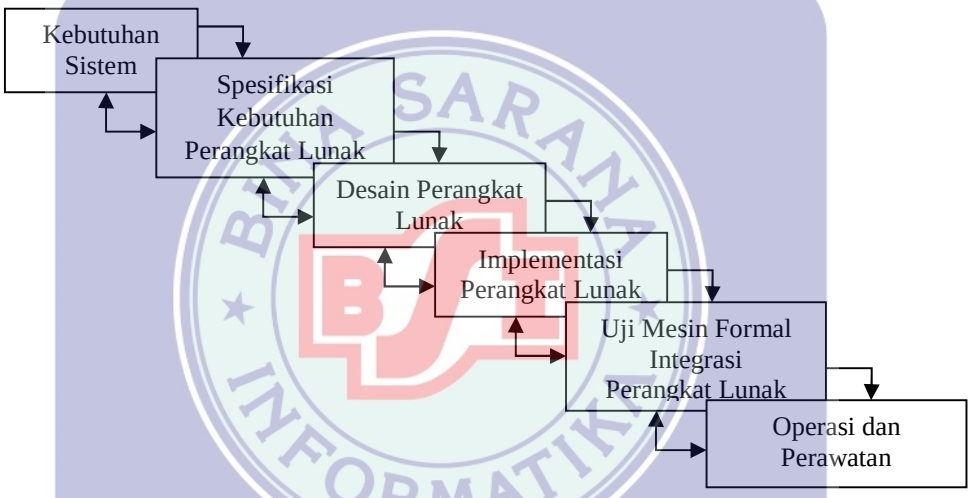
Elemen-elemen penyusunan *database* antara lain adalah :

1. Tabel, merupakan kumpulan *record* dengan format *field* yang sama. Satu tabel biasanya mempersentasikan data satu objek maupun satu kejadian yang terjadi dalam sebuah sistem.
2. *Field* / Kolom, merupakan bagian terkecil dari tabel yang digunakan untuk menyimpan informasi item.
3. *Primary key*, merupakan suatu *field* yang nilainya unik dan digunakan sebagai kunci yang membedakan *record* satu dengan *record* yang lainnya.
4. *Relationship*, hubungan antara satu tabel dengan tabel yang lainnya.
5. *Record*/ baris, merupakan suatu *field* yang berhubungan erat menggambarkan satu informasi.
6. *Query*, digunakan untuk menyaring dan menampilkan data memenuhi kreteria tertentu dalam satu atau lebih.

D. Model Penggembangan Perangkat Lunak

Simarmata (2010:53) pada tahun 1960-an, proyek pengembangan perangkat lunak merupakan pekerjaan yang sangat memakan biaya dan waktu karena pengembangan perangkat lunak ini difokuskan pada perencanaan dan pengendalian (Basili dan Musa, 1991). Kemunculan model air terjun adalah

untuk membantu mengatasi kerumitan yang terjadi akibat proyek-proyek pengembangann perangkat lunak (Boehm,1976). Seperti yang terlihat pada gambar dibawah ini, sebuah model air terjun memacu tim pengembang untuk merinci apa yang seharusnya perangkat lunak lakukan (mengumpulkan dan menetukan kebutuhan sistem) sebelum sistem tersebut dikembangkan.



Sumber: (Simarmata:2010)

Gambar II.1.Waterfall Model

Kemudian, model ini memungkinkan pemecahan misi pengembangan yang rumit menjadi beberapa langkah logis (desain, kode, pengujian dan seterusnya) dengan beberapa langkah yang pada akhirnya akan menjadi produk akhir yang siap pakai. Untuk memastikan bahwa sistem biasa dijadikan, setiap langkah akan membutuhkan validasi, masukan dan kriteria yang ada.

Banyak metode yang sering digunakan dalam pengembangan sistem seperti *Prototyping*, Metode *Waterfall* dan masih banyak lagi yang lainnya. Tulisan ini hanya membahas metode *waterfall*.

Metode *Waterfall* ini sebenarnya adalah "*Linier Sequental Model*", yang sering juga disebut dengan "*Classic life cycle*" atau model *waterfall*. Metode ini muncul pertama kali sekitar tahun 1970 sehingga sering dianggap kuno, tetapi merupakan model atau metode yang paling banyak dipakai di dalam *Software Enginnering* (SE).

"Metode *Waterfall* ini melakukan pendekatan secara sistematis dan urut mulai dari *level* kebutuhan sistem lalu menuju ke setiap analisis, *design*, *coding*, *testing/verification*, dan *maintenenance*" (Muharto 2016:105).

1. Analisa dan definisi kebutuhan. Layanan, batasan, dan tujuan sistem ditentukan melalui konsultasi dengan *user* atau pemakai.
2. Perancangan sistem dan perangkat lunak. Proses perancangan sistem membagi persyaratan dalam sistem perangkat keras atau perangkat lunak. Kegiatan ini menetukan arsitektur sistem secara keseluruhan. Perancangan melibatkan identifikasi dan deskripsi abstraksi sistem perangkat lunak yang mendasar.
3. Implementasi dan pengujian unit. Pada tahap ini, perancangann atau unit program. Pengujian ini melibatkan verifikasi atau unit program. Pengujian ini melibatkan verifikasi bahan setiap unit telah memenuhi sepesifikasinya.

4. Integrasi dan pengujian sistem. Unit program atau program individu di integrasi dan diuji sebagai sistem yang lengkap untuk menjamin bahwa kebutuhan sistem telah dipenuhi.
5. Operasi dan pemeliharaan, yaitu mengoperasikan program di lingkungannya dan melakukan pemeliharaan. Biasanya ini merupakan fase siklus hidup yang paling lama. Pemeliharaan mencakup koreksi dari berbagai *error* yang tidak ditemukan pada tahap-tahap sebelumnnya, melakukan perbaikan atas implementasi unit sistem dan pengembangan layanan sistem, dan persyaratan-persyaratan baru ditambahkan.

2.2. Teori Pendukung

Untuk lebih memahami isi dan perancangan web pada tugas akhir ini, maka dibutuhkan beberapa pengetahuan mengenai definisi serta uraian yang berkaitan dengan teori pendukung, sebagai berikut:

A. Struktur Navigasi

Menurut Simarmata (2010:309) mengemukakan "navigasi yang ada pada situs web atau aplikasi menujukkan sesuatu yang penting dan menjadi kata kunci *usability* aplikasi". Tersesat di dalam "*sindrom hyperspace*" pada navigasi searah memang harus dihindari. Oleh karena itu, pengembang perlu menyampaikan suatu model mental dari struktur navigasi yang cepat dan membiarkan para pengguna untuk "menghafal peta situs". Struktur dasar logika yang jelas dan didukung oleh suatu peta situs, seperti umpan balik tetap pada posisi yang di dalam struktur ("Di mana saya?"), informasi yang jelas tentang isi pada halaman yang ada("Apa yang bisa saya lakukan atau

temukan di sini?"), dan item-item yang dapat dicapai pada langkah interaksi yang berikutnya ("Ke mana saya pergi?") ada ramuan-ramuan yang paling penting untuk sebuah sistem dialog dan navigasi yang dirancang dengan baik.

Ada beberapa aspek yang perlu dipertimbangkan ketika akan merancang situs *Web* (simarmata, 2010:310), yaitu :

1. *Usabilitas*

   Usabilitas merupakan hal yang sangat penting dalam merancang *web*. Memiliki sebuah situs *Web* dinamis yang tampak profesional dan bagus memang sangat baik, namun jika *web* tersebut membutuhkan waktu yang sangat lama untuk memuat sebuah artikel atau pengunaan navigasinya sangat rumit, tidak heran jika pengguna akan keluar dari situs *web* anda dan tidak akan kembali lagi. Perlu diketahui bahwa pada umumnya pengguna ingin mendapatkan informasi secara cepat, meskipun tampialan situs *web*-nya biasa saja. Jika pencarian informasinya terlalu lama, para pengguna akan langsung menutup halaman *web* tersebut. Ingat, jangan mengorbankan aspek *usabilitas* dalam mendesain situs *web.*

2. Navigasi

   Navigasi juga menjadi hal yang sangat penting dalam sebuah situs *web* yang berfungsi untuk membantu pengguna menjelajai situs *web* untuk mencari informasi yang diinginkan secara mudah.

   Navigasi yang baik akan mencerminkan struktur situs *web* yang baik.

   Navigasi pada sebuah situs *web* yang tampil pada menu dan tautan merupakan petunjuk bagi pengunjung mengenai halaman-halaman

dalam situs *web* anda, jika menu-menu tautan yang tampil secara terstruktur. Pengunjung pasti akan merasa kesal apabila tidak mendapatkan halaman situs *web* yang di cari hanya karena navigasi terlalu rumit.

B. *Entity Relationship Diagram*

Menurut Fatta (2009:27) "ERD (*Entity Relatonship Diagram*) adalah suatu model jaringan yang menggunakan susunan data yang disimpan dalam sistem secara abstrak". ERD merupakan jaringan data yang menekankan pada struktur dan hubungan antar data. ERD juga memperlihatkan hubungan antara *store* pada DFD. Diagram hubungan entitas, atau yang lebih dikenal dengan E-R, adalah notasi grafik dari sebuah model data atau sebuah model jaringan yang menjelaskan tentang data yang disimpan (*storage data*) dalam sistem abstrak. Diagram hubungan entitas tidak menyatakan bagaimana memanfaatkan data, membuat data, mengubah data, dan menghapus data.

ERD menjadikan salah satu pemodelan data konseptual yang paling sering digunakan dalam proses pengembangan basis data bertipe relasional. Penggunaannya sangat luas di akibatkan beberapa faktor, yaitu kemudahan penggunaan secara luas *Commputer Aided Software Engineering* (CASE), dukungan konsep matematika (kalkulus relasional) yang tangguh, hubungan entitas antar entitas merupakan konsep pemodelan alamiah yang sesuai dengan keadaan dunia nyata.

Model E-R sering digunakan sebagai sarana komunikasi antara perancang basis data dan pengguna sistem selama tahap analisa dari proses

pengembangan basis data dalam kerangka pengembangan sistem informasi secara utuh. Model E-R digunakan untuk mengkonstruksikan model data konseptual, yang mencerminkan struktur data dan bastasan dari basis data, yang mandiri dari perangkat lunak pengelolah basis data (DBMS) dan berhubungan erat dengan model data yang langsung bisa digunakan untuk mengimplementasikan basis data secara logika maupun secara fisik dengan DBMS yang dipilih pada tahap implementasi.

Model data E-R pertama kali diperkenalkan oleh Chen (1976) pada artikelnya yang mendiskusikan konstruksi utama dari model E-R, hubungan antar entitas (*relationship*) serta atribut-atribut yang bersesuaian dengan tiap entitas. Model yang diperlukan oleh Chen kemudian diperluas dan dikembangkan oleh Teorrey, yang fry (1986), serta Story (1991) saat ini model E-R masih berkembang, namun tidak ada notasi baku untuk pemodelan E-R.

ERD adalah salah satu diagram untuk menggambarkan desain konseptual dari model konseptual suatu basis data relasional. ERD juga merupakan gambaran yang merelasikan antara objek yang satu dengan objek yang lain dari objek di dunia nyata yang sering dikenal dengan hubungan atara entitas. Sebagai contoh jika membuat ERD dari sistem perpustakaan maka bahan sebagai objek ERD bisa berupa anggota, buku, pinjaman, pengembalian dan sebagainya. ERD terdiri dari 3 komponen utama, yaitu:

1. Entitas (*Etity*)

   Entitas adalah suatu objek di dunia nyata yang dapat dibedakan dengan objek lainnya. Objek tersebut dapat berupa orang, benda ataupun hal

lainnya. Entitas digambarkan dalam bentuk persegi seperti gambar di atas, terlihat dari jenisnya entitas terbagi atas dua yaitu:

a. Entitas Kuat (*strong Entity*)

   Entitas kuat adalah entitas yang dapat berdiri sendiri tidak bergantung pada entitas lainnya, entitas kuat memiliki atribut key dan entitas kuat digambarkan tunggal. Contoh entitas kuat adalah entitas pegawai.

b. Entitas Lemah (*Weak Entitiy*)

   Entitas lemah adalah entitas yang tidak dapat brdiri sendiri. Entitas lemah merupakan hasil dari pembentukan entitas kuat, entitas lemah tidak memiliki seba atribut *key* dan entitas lemah digambarkan sebagai kontak persegi panjang bergaris ganda. Jika entitas kuat yang membentuk entitas lemah dihapus maka secara otomatis entitas lemah akan terhapus. Contoh entitas lemah adalah entitas pegawai kontrak, pegawai tetap.

2. Atribut (*attribute*)

Atribut merupakan semua informasi yang berkaitan dengan entitas. Atribut sering dikenal dengan *property* dari satu entitas atau objek. Atribut digambarkan dalam bentuk lingkaran elips. Macam-macam atribut:

a. Atribut Sederhana (*Simple Attribute*)

   Atribut sederhana adalah atribut yang nilainya tidak dapat dibagi lagi menjadi banyak atribut yang lebih kecil. Contoh atribut sederhana harga seperti contoh gambar di bawah ini.

b. Atribut Komposit (*Composite Atribute*)

Atribut komposit adalah atribut gabungan yang nilainya dapat dipecah menjadi bagian yang lebih kecil. Atau sering disebut atribut yang terdiri dari beberapa atribut kecil di dalamnya.

c. Atribut bernilai tunggal (*Singgle Values Attribute*)

Atribut bernilai tunggal adalah jenis atribut yang nilainya hanya satu dari satu entitas. Contoh atribut bernilai tunggal adalah tanggal_lahir dari entitas mahasiswa. Telah ada bisa dipastikan bahwa setiap mahasiswa mempunyai satu tanggal_lahir.

d. Atribut Bernilai Banyak (*multivalues Attribute*)

Atribut bernilai banyak adalah jenis atribut yang nilainya lebih dari satu dalam satu entitas tertentu. Contoh atribut bernilai banyak adalah hobi dimungkinkan bahwa mahasiswa memilik lebih dari satu.

e. Atribut Turunan (*Derive Attribute*)

Atribut turunan adalah jenis atribut yang nilainya diperoleh dari atribut yang lain. Contoh atribut turunan adalah masa_bakti dari entitas pegawai.

Atribut masa_bakti akan muncul nilainya ketika atribut tanggal_masuk_kerja sudah ada nilainya. Pada dasarnya atribut masa bakti tidak akan dijadikan suatu kolom. Atribut masa_bakti akan muncul dengan bantuan query.

f. Atribut Identitas (*Key Atrribute*)

Atribut identitas adalah atribut yang dijadikan sebagai kunci pada suatu *table*. Sifat atribut identitas ini unik, tidak ada yang menyamai, atribut identitas terdiri dari beberapa jenis yaitu:

a. *Super Key*

Super key adalah suatu atribut atau kumpulan atribut yang secara unik mengidentifikasi sebuah baris di dalam relasi atau himpunan dari satu atau lebih entitas yang dapat digunakan untuk mengidentifikasi secara unik sebuah entitas dalam set entitas.

b. *Cadidate key*

Cadidate key adalah atribut yang menjadi determainan yang dapat dijadikan identitas dari pada sebuah relasi. Biasanya *super key* minimum.

c. *Primary Key*

Primary key adalah *cadidate key* yang tidak terpilih sebagai *primary key* atribut untuk menggantikan kata kunci utama

d. *Alternative Key*

Alternative key adalah *cadidate key* yang tidak terpilih sebagai *primary key* atau atribut untuk menggantikan kunci utama.

e. *Foreign key*

Foreign key adalah atribut dengan domain yang sama yang menjadi kunci utama sebuah relasi, tetapi pada relasi lain atribut tersebut sebagai atribut biasa.

f. *Composite Key*

Composite key adalah kunci yang terdiri dari dua atribut atau lebih. Atribut-atribut tersebut jika berdiri sendiri tidak menjadi identitas baris, tetapi bia digambarkan menjadi satu kesatuan akan dapat mengidentifikasi secara unik.

3. Tipe Relasi

Gambar belah ketupat merupakan pelambangan relasi antar entitas atau sering disebut kerelasian. Ada dua macam penggambaran relasi yaitu relasi kuat dan relasi lemah. Relasi kuat adalah untuk menghubungkan antar entitas kuat sedangkan relasi lemah untuk menghubungkan atar entitas kuat dengan entitas lemah.

Ada tiga macam relasi menurut derajatnya, yaitu:

a. *unary* adalah relasi yang menghubungkan entitas yang sejenis.
b. *binary* adalah relasi yang menghubungkan entitas yang tidak sejenis.
c. *ternary* adalah relasi yang menghubungkan lebih dari dua entitas yang tidak sejenis.

4. Derajat Kardinalitas

Derajat kardinalitas merupakan penjabaran dari hubungan antar entitas. Derajat kardinalitas dibagi atas tiga bagian yaitu:

a. Derajat kardinalitas *One to One*

Derajat kardinalitas *one to one* terjadi jika satu entitas X hanya berelasi dengan satu entitas Y, ataupun sebaliknya. Sebagai contoh satu pegawai satu hanya memiliki satu pendamping.

Sumber: (Yanto:2016)

Gambar II.2. Derajat Kardinalitas One to One

b. Derajat kardinalitas *One to Many*

Derajat kardinalitas *one to many* terjadi jika satu entitas X berelasi dengan banyak entitas Y, ataupun sebaliknya. Sebagai contoh satu dosen menampung banyak mahasiswa.

Sumber: (Yanto:2016)

Gambar II.3. Derajat Kardinalitas One to Many

c. Derajat Kardinalitas *Many to Many*.

Derajat kardinalitas *many to many* terjadi jika banyak entitas X berelasi dengan banyak entitas Y, jika atau pun sebaliknya.

Sumber: (Yanto:2016)

Gambar II.4. Derajat Kardinalitas Many to Many

C. LRS (*Logical Record Structure*)

Menurut Wulandari (2013:15) "*Logical Record Structure* dibentuk dengan nomor dan tipe *record*, beberapa tipe *record* digambarkan oleh kotak-kotak persegi panjang dan dengan nama yang unik ". LRS terdiri dari *link-link* diantara tipe *record*. *Link* ini menunjukan arah dari satu tipe *record* lainnya. Banyak *link* dari LRS yang diberi tanda *field-field* yang kelihatan ada kedua *link type record*..

Dalam kaitannya dengan konversi ke LRS, untuk perubahan yang terjadi adalah mengikuti aturan-aturan berikut:

1. Setiap *entitas* diubah ke bentuk kotak dengan nama *entitas*, berada diluar kotak dan atribut berada didalam kotak.
2. Seluruh *relationship* kadang disatukan, dalam sebuah kotak bernama *entitas*, kadang sebuah kotak bersama-sama dengan *entitas*. Kadang disatukan dalam sebuah kotak sendiri.
3. Konversi LRS ke relasi table adalahh bentuk pernyataan data secara grafis dimensi, yang terdiri dari kolom dan baris. Relasi adalah bentuk visual dari sebuah *file*, dan tiap *tuple* dalam sebuah *field*, atau yang daam bentuk lingkaran diagram *entity relationship* dikenal dengan sebutan atribut.

D. Pengujian *Web*

Metode pengujuan yang merupakan akhir dari rangkuman perkuliahan metode penelitian, namun bukanlah akhir dari materi pengembangan melainkan landasan yang masih terbuka untuk di teliti lebih lanjut kembali.

Metode pengujian dalam penelitian rekayasa merupakan aspek yang sangat penting karena melalui pengujian inilah reliabilitas hasil rekayasa dapat di dekati lebih baik. Selain faktor kualitas lainya.

Menurut Martudi (2014:65) “Pengujian merupakan satu elemen dari verifikasi (untuk memastikan bahwa perangkat lunak secara tepat mengimplementasikan suatu fungsi tertentu). Dan validasi (untuk memastikan perangkat lunak dapat diterlusuri hingga ke persyaratan yang diminta pelanggan)”.

Fatta (2007:169) mengemukakan “pengujian sistem merupakan proses mengeksekusi sistem perangkat lunak untuk menentukan apakah ada sistem perangkat lunak tersebut cocok dengan spesifikasi sistem dan berjalan sesuai dengan lengkungan yang diinginkan”. Pengujian sistem sering diasosiasikan dengan pencarian *bug*, ketidak sempurnaan program, kesalahan pada baris program yang menyebabkan kegagalan pada eksekusi sistem perangkat lunak. Menemukan dan menghilangkan ketidaksempurnaan program ini disebut *debugging*, yang berbeda debgan pengujian sistem yang berfokus pada pengidentifikasian adanya ketidaksempurnaan ini. Kesalahan sistem perangkat lunak bisa sangat bervariasi, mulai dari *output* yang salah, sistem yang *crash*, sampai pada sistem yang menggunakan memori terlalu banyak atau sistem yang berjalan terlalu lambat. *Bug-bug* seperti ini terjadi karena kode program yang dieksekusi belum diuji, urutan program yang dieksekusi belum diuji kombinasi dari nilai *input* yang belum diuji, atau lingkungan program-program belum diuji.

Jika struktur kendalai antar modul sudah terbukti bagus, maka pengujian yang tak kalah pentingnya adalah pengujian unit, pengujian unit digunakan untuk menguji modul untuk menjamin setiap modul menjalankan fungsinya dnengan baik. Ada dua metode untuk melakukan unit *testing*, yaitu:

1. *Black Box Testing*

   Terfokus pada apakah unit program memenuhi kebutuhan (*requirment*) yang disebutkan dalam spesifikasi. "*Black box testing*, cara pengujian hanya dilakukan dengan menjalankan atau mengeksekusi unit atau modul, kemudian diamati apakah hasil dari unit itu sesuai dengan proses bisnis yang diinginkan" Fatta (2007:172). Jika ada unit yang tidak sesuai *output*-nya maka untuk menyelesaikannya diteruskan pada pengujian yang kedua, yaitu *white box testing*.

2. *White Box Testing*

   Fatta (2007:172) mengemukakan "*White box testing* adalah cara pengujian dengan melihat ke dalam modul untuk meneliti kode-kode program yang ada, dan menganalisis apakah ada kesalahan atau tidak". Jika ada pada modul yang menghasilkan *output* yang tidak sesuai dengan proses bisnis yang dilakukan, maka baris-baris program, variabel, dan parameter yang terlibat pada unit tersebut akan dicek satu persatu dan diperbaiki, kemudian di-*compline* ulang.

Integration Testing. Setelah unit testing selesai maka pengujian berikutnya adalah pengujian interaksi dari modul-modul yang menyusun sistem informasi untuk menjamin bahwa mereka bekerja dengan baik. *Integration* tes terdiri dari seragkaian tes sebagai berikut:

a. Ujicoba antarmuka

   Ujicoba setiap fungsi dari antarmuka.

b. Ujicoba skenario pengguna

   Pastikan setiap skenario berjalan dengan baik.

c. Ujicoba aliran data

   Uji setiap proses dalam langkah per langkah.

d. Ujicoba sistem antarmuka

   Pastikan data mengalir antar proses.