BAB II

LANDASAN TEORI

2.1. Konsep Dasar *Web*

Dalam pembuatan tugas akhir ini diperlukan teori-teori yang sesuai untuk mendukung kemudahan dalam mempelajari dan merancang *web* yang diinginkan. Sehingga, aplikasi *web* ini dapat difungsikan secara maksimal dan mudah digunakan oleh pengguna.

2.1.1. *Website*

Website merupakan kumpulan halaman informasi yang berupa informasi teks, gambar, animasi, suara, atau gabungan dari semuanya, bersifat statis maupun dinamis yang saling berkaitan dan terhubung dengan jaringan-jaringan halaman. (Bekti, 2015:7).

Menurut Abdulloh (2016: 1) "*Website* merupakan halaman-halaman yang berisi informasi yang ditampilkan oleh *browser*, seperti *Mozilla Firefox*, *Google Chrome*, atau yang lainnya".

Ada 2 (dua) tipe halaman *web*, statis dan dinamis. *Web* statis adalah halaman *web* yang mempunyai tampilan yang sama saat diakses oleh berbagai macam pengguna sehingga informasi yang ditampilkan kepada pengguna selalu sama karena tidak berubah. Sedangkan, *web* dinamis adalah *web* yang mampu menampilkan hasil tampilan mengikuti kondisi penggunanya atau kondisi *server*.

1. *Internet*

   Menurut Ahmadi dan Hermawan (2013: 68) "*Internet* adalah komunikasi jaringan komunikasi global yang menghubungkan seluruh komputer di dunia meskipun berbeda sistem operasi dan mesin".

   Azis (2013: 6) menjelaskan bahwa "*Internet* merupakan sebuah jaringan (*Internet Protocol*) yang terdiri dari beberapa komputer yang sudah terkoneksi ke dalam jaringan global".

   Internet merupakan jaringan komputer yang dibentuk oleh Departemen Pertahanan Amerika Serikat pada tahun 1969, melalui proyek ARPA yang disebut ARPANET (*Advanced Research Project Agency Network*), di mana mereka mendemonstrasikan bagaimana penggunaan *hardware* dan *software* komputer yang berbasis UNIX, di mana dengan menggunakan proyek tersebut kita bisa melakukan komunikasi dalam jarak yang tidak terhingga melalui saluran telepon.

2. *Web Browser*

   Menurut Sibero (2013: 12) "*Web Browser* adalah aplikasi perangkat lunak yang digunakan untuk mengambil dan menyajikan sumber informasi *web*". *Web browser* pertama kali dimulai pada tahun 1991 saat Tim Berners-Lee membuat aplikasi *web browser* pertama pada komputer NeXT dengan nama *World Wide Web Browser*, kemudian di tahun 1993 NCSA (*National Center Supercomputing Application*) mengembangkan *web browser* grafis bernama NCSA *Mosaic*, yang kemudian dilanjutkan pada tahun 1994 merilis *Netscape Navigator* dan pada tahun 1998 nerubah menjadi *Mozilla Firefox*.

3. *Web Server*

   Web server merupakan perangkat lunak dalam *server* yang berfungsi menerima permintaan halaman *web* dari klien melalui *browser* dan mengirimkan hasil berbentuk halaman-halaman *web* yang berbentuk dokumen HTML. (Solichin, 2016: 7).

   Menurut Mundzir (2014: 10) "*Web server* merupakan tempat berkumpulnya segala kode-kode pemrograman dan kode lainnya yang mana semua program tersebut bekerja sesuai dengan fungsinya masing-masing". Jadi, bisa diibaratkan *web server* adalah sebuah gedung dengan kode-kode program sebagai para karyawan dalam gedung tersebut.

4. *Domain*

   Nilasari (2014: 21) mengatakan bahwa "*Domain* merupakan alamat unik di dunia maya yang digunakan untuk mengidentifikasi sebuah *website*." *Domain* bisa disebut juga sebagai alamat *website* karena *domain* akan memudahkan kita dalam mengakses sebuah *website*. Contoh *domain*: bsi.ac.id, google.com, wikipedia.org, kpu.go.id dan lain-lain.

5. *Web Hosting*

   Menurut Nilasari (2014: 27) "*Web Hosting* adalah sebuah layanan *hosting internet* yang memberikan akses pada individu atau organisasi". *Web hosting* adalah salah satu syarat agar *website* bisa *online* dan dapat diakses *internet*. Ukuran yang digunakan pada *web hosting* adalah kapasitas dan *bandwith*. Kapasitas adalah kemampuan *web hosting* dalam menyimpan data-data *website*. Sedangkan *bandwith* adalah jumlah *volume* data yang diperbolehkan untuk diakses dari *web hosting* setiap bulannya.

2.1.2. Bahasa Pemrograman

Bahasa pemrograman merupakan bahasa yang dimengerti oleh komputer. Terdapat banyak bahasa pemrograman yang memiliki fungsi yang berbeda. Dalam membuat sebuah *website* ada banyak jenis bahasa pemrograman yang dapat digunakan, dalam penulisan tugas akhir ini penulis menggunakan beberapa bahasa pemrograman yaitu:

1. HTML *(Hypertext Markup Languange*)

   Menurut Abdulloh (2016: 2) "*Hypertext Markup Languange,* yaitu skrip yang berupa tag-tag, untuk membuat dan mengatur struktur *website*."

   Winarno, dkk (2013: 1) mengatakan "HTML merupakan singkatan dari *Hypertext Markup Languange*, artinya bahasa ini adalah bahasa *markup* untuk memformat konten halaman *web*".

   Tugas HTML dalam mebangun *website* yaitu:

   a. Menentukan layout *website*
   b. Memformat teks dasar, seperti pengaturan paragraf dan *format font*
   c. Membuat list
   d. Membuat tabel
   e. Menyisipkan gambar, video, dan audio
   f. Membuat link
   g. Membuat formulir

2. PHP (*Hypertext Preprocessor*)

   Mundzir (2014: 7) mengatakan "*Hypertext Preprocessor* yaitu bahasa pemrograman *universal* untuk penanganan pembuatan dan pengembangan sebuah situs web dan bisa digunakan bersamaan dengan HTML".

Rasmus Lerdorf merupakan pembuat PHP pertama kali pada tahun 1995. Dulu, PHP merupakan singkatan dari *Personal Home Page* atau lebih dikenal sebagai situs bernama *Form Interpreted* (FI). *Form Interpreted* (FI) memiliki wujud berupa sekumpulan skrip yang digunakan untuk mengolah data formulir dari *web*. Tahap selanjutnya, Rasmus merilis kode sumber tersebut untuk umum dan menamakannya PHP/FI.

Pada Juni 2004, *Zend* merilis PHP 5.0. Adanya PHP 5.0 ini ditandai dengan masuknya model pemrograman berorientasi objek ke dalam PHP untuk menjawab perkembangan bahasa pemrograman ke arah paradigma berorientasi objek.

3. *Javascript*

Menurut Atmoko (2018: 18) "*Javascript* merupakan bahasa pemrograman *web Client Side. Client Side Programming Language* adalah bahasa pemrograman yang pemrosesannya dilakukan oleh *client*/*browser* seperti *Google Chrome* dan *Mozilla Firefox*".

Abdulloh (2016: 3) menjelaskan bahwa "Peran *javascript* dalam membuat *website* adalah memberikan efek animasi yang menarik dan interaktivitas dalam penanganan *event* yang dilakukan oleh pengguna *website*".

4. *Jquery*

Rohingun (2015: 1) mengemukakan bahwa *"Jquery* merupakan salah satu dari sekian banyak *javascript library* yaitu kumpulan fungsi *javascript* yang siap pakai, sehingga mempermudah dan mempercepat kita dalam membuat kode *javascript".*

Menurut Bekti (21015: 59) "*Jquery* merupakan salah satu librari yang membuat program *web* di sisi klien tidak seperti program *JavaScript* bias, yang harus secara eksplisit disisipkan pada dokumen *web*".

Keunggulan menggunakan *jquery* dibandingkan dengan cara memanggil fungsi-fungsi yang disediakan oleh *jquery. Jquery* pertama kali dirilis tahun 2006 oleh John Resig. *Jquery* menjadi sangat populer hingga telah banyak digunakan pada *website* termasuk *website* kelas dunia seperti *goole, amazone, twitter,* dan lain-lain.

5. CSS (*Cascading Style Sheet*)

Bekti (2015: 47) mengatakan bahwa "CSS (*Cascading Style Sheets*) merupakan salah satu bahasa pemrograman *web* yang digunakan untuk mempercamntik halaman *web* dan mengendalikan beberapa komponen dalam sebuah *web* sehingga lebih terstruktur dan seragam".

Menurut Abdulloh (2016: 2) "CSS singkatan dari *Cascading Style Sheets*, yaitu skrip yang digunakan untuk mengatur desain *website*. Fungsi CSS adalah memberikan pengaturan yang lebih lengkap agar struktur *website* yang dibuat dengan HTML terlihat lebih rapi dan elegan".

2.1.3. Basis Data

Solichin (2016: 84) mengatakan "Basis data atau *database* kumpulan informasi yang disimpan di dalam komputer secara sistematik sehingga dapat diperiksa menggunakan suatu program komputer untuk memperoleh informasi dari basis data itu".

Menurut Enterprise (2014: 1) "*Database* adalah suatu aplikasi yang menyimpan sekumpulan data". *Relational Database Management Systems*

(RDBMS) digunakan untuk menampung dan mengatur data yang begitu banyak karena semua data disimpan dalam tabel-tabel yang berbeda dan dihubungkan berdasarkan relasinya dengan menggunakan *primary key* dan *foreign key*.

1. *MySQL (My Structure Query Languange)*

   Menurut Solichin (2016: 85) "*MySQL* adalah sebuah perangkat lunak sistem manajemen basis data SQL (bahasa Inggris: *database management system*) atau DBMS yang *multithread*, *multi-user*".

   Enterprise (2014: 2) mengemukakan bahwa "*MySQL* adalah *Relational Database Management Systems* (RDBMS) yang cepat dan mudah digunakan, serta sudah banyak digunakan untuk berbagai kebutuhan". *MySQL* dikembangkan oleh *MySQL* AB Swedia. Beberapa kelebihan menggunakan *MySQL* yaitu:

   a. Berlisensi *open source*, sehingga dapat digunakan secara gratis.
   b. Program yang *powerful* dan menyediakan fitur yang lengkap.
   c. Menggunakan bentuk standar bahasa data SQL.
   d. Dapat bekerja dengan banyak sistem operasi dengan banyak bahasa pemrograman.
   e. Bekerja dengan cepat dan baik, meskipun datanya banyak.
   f. Sangat mudah digunakan dengan PHP untuk pengembangan aplikasi *web*.

2. *PhpMyAdmin*

   Menurut Mandar (2017: 11) "*phpMyAdmin* adalah salah satu dari tiga aplikasi yang *include* pada satu paket *software web server* seperti *Wampserver* atau *Xampp*".

Abdulloh (2016: 6) mengatakan bahwa "*phpMyAdmin* merupakan aplikasi berbasis *web* yang digunakan untuk membuat *database MySQL* sebagai tempat untuk menyimpan data-data *website*".

2.1.4. *Software* Pendukung

Dalam pembuatan tugas akhir ini penulis menggunakan beberapa *software* yang mendukung pembuatan *website* tugas akhir ini:

1. *Adobe Dreamweaver CS6*

   Menurut Wahana Komputer (2013: 2) "*Adobe Dreamweaver CS6* merupakan versi terbaru dari *Adobe Dreamweaver CS5*". Aplikasi *Adobe Dreamweaver CS6* memberikan tampilan yang lebih baik dan tentu saja semakin mudah dalam penggunaannya. Aplikasi ini mengintegrasikan beragam fitur untuk memenuhi kebutuhan pengembangan *website*, termasuk pembuatan halaman *web* dan pengelolaannya.

2. *Sublime Text*

   Supono dan Putratama (2016: 14) menjelaskan bahwa "*Sublime text* merupakan perangkat lunak *text editor* yang digunakan untuk membuat atau meng-*edit* suatu aplikasi". *Sublime text* mempunyai tampilan yang ringan, elegan, dan mempunyai fitur *pugin* tambahan yang memudahkan *programmer*.

3. *Bootstrap*

   Bootsrap merupakan sebuah *framework* CSS yang mudah digunakan. Abdulloh (2016: 157) menjelaskan bahwa "Dengan menggunakan *bootstrap*, proses desain *website* tidak dibuat dari nol, sehingga proses desain *website* lebih cepat dan mudah".

Sejak tahun 2012 *bootstrap* dilengkapi oleh fitur *resposive*, dengan fitur *resposive*, *website* dapat dilihat dalam berbagai ukuran layar, seperti *smartphone* atau tablet dengan desain tetap teratur dan mengikuti ukuran layar. Saat penulisan tugas akhir ini *bootstrap* telah sampai pada *versi* 4.

4. *Xampp*

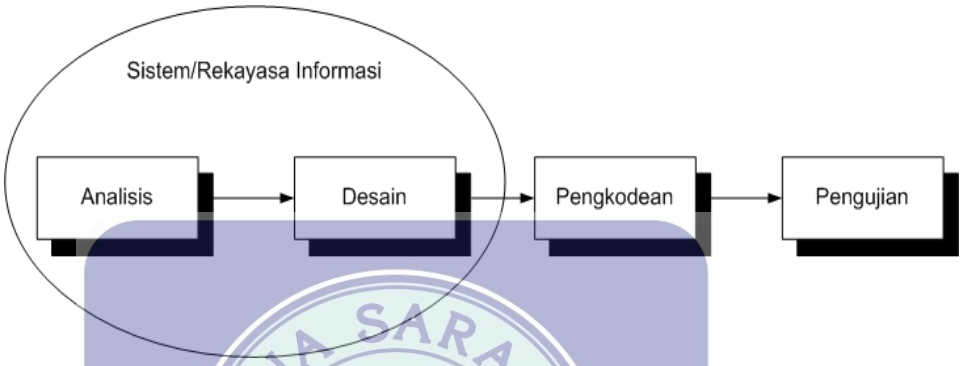
Abdulloh (2016: 7) mengatakan bahwa "*Xampp* adalah salah satu paket *installer* yang berisi *Apache* yang merupakan *web server* tempat menyimpan *file-file* yang diperlukan *website*, dan *PhpMyAdmin* sebagai aplikasi yang digunakan untuk perancangan *database MySQL*".

2.1.5. Model Pengembangan Perangkat Lunak

Menurut Sukamto dan Shalahuddin (2016: 25) menjelaskan bahwa "pada awal pengembangan perangkat lunak, para pembuat program (*programmer*) langsung melakukan pengkodean perangkat lunak tanpa menggunakan prosedur atau tahapan pengembangan perangkat lunak".

SDLC atau *Software Development Life Cycle* atau sering disebut juga *System Development Life Cycle* adalah proses mengembangkan atau mengubah suatu sistem perangkat lunak dengan menggunakan model-model dan metodologi yang digunakan orang untuk mengembangkan sistem-sistem perangkat lunak sebelumnya (berdasarkan *best practice* atau cara-cara yang sudah teruji baik). Seperti halnya proses metamorfosis pada kupu-kupu, untuk menjadi kupu-kupu yang indah maka dibutuhkan beberapa tahap untuk dilalui, sama halnya dengan membuat perangkat lunak, memiliki daur tahapan yang dilalui agar menghasilkan perangkat lunak yang berkualitas.

Dalam perancangan aplikasi pada tugas akhir ini penulis menggunakan SDLC model *Waterfall*. Menurut Sukamto dan Shalahudin (2016: 28) "model pendekatan *waterfall* menyediakan pendekatan alur hidup perangkat lunak secara sekuensial atau terurut dimulai dari analisis, desain, pengodean, pengujian dan tahap pendukung (*support*)". Berikut adalah gambar metode *waterfall*:



Sumber: Sukamto dan Shalahudin (2016: 29)

Gambar II.1.
Gambar Metode *Waterfall*

1. Analisis Kebutuhan Perangkat Lunak

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

2. Desain Perangkat Lunak

   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 pengodean. Tahap ini mentranslasi kebutuhan perangkat lunak dari tahap analisis kebutuhan ke representasi 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 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.

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

   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.

2.1.6. Pengertian Pinjaman

Menurut Hasibuan dalam Setiawan dan Rinawati (2015: 40) pinjaman adalah “suatu pemberian jasa oleh suatu pihak terhadap pihak lain atas dasar kepercayaan”. Sedangkan menurut Kasmir dalam Sudrajat dan Sudrajat menyatakan bahwa “kredit adalah hak untuk menerima pembayaran atau kewajiban untuk melakukan pembayaran pada waktu diminta, atau pada waktu yang akan datang karena penyerahan barang-barang sekarang”.

2.1.7. Pembayaran

Menurut Drebin dalam Setiawan dan Rinawati (2015: 41) “Pembayaran uang tunai periodik sebagai pembayaran angsuran yang besarnya telah ditentukan sebelumnya atau ditentukan besar kecilnya yang tergantung pada lamanya besar angsuran”.

Pembayaran yang lancar merupakan salah satu syarat mencapai kestabilan perputaran uang yang terjadi di sebuah badan usaha peminjaman. Maka sistem pembayaran perlu diatur dan dijaga keamanan serta kelancarannya.

2.1.8. Koperasi

Menurut Wibowo dan Subagyo (2017: 5) “Koperasi adalah badan usaha yang beranggotakan orang-seorang atau badan hukum koperasi dengan melandaskan kegiatannya berdasarkan prinsip koperasi sekaligus sebagai gerakan ekonomi rakyat yang berdasar atas asas kekeluargaan”.

Koperasi bertujuan memajukan kesejahteraan anggota pada khususnya dan masyarakat pada umumnya serta ikut membangun tatanan perekonomian nasional dalam rangka mewujudkan masyarakat yang maju, andil, dan makmur berlandaskan Pancasila dan Undang-Undang Dasar 1945.

2.1.9. Penelitian Terkait

Ada beberapa penelitian mengenai aplikasi peminjaman dan pembayaran yang dibuat oleh beberapa peneliti seperti:

1. Sardiarinto (2013) melakukan penelitian dengan judul “Aplikasi Sistem Pendukung Keputusan Kelayakan Peminjaman Kredit Nasabah Koperasi Berbasis *Android*”. Pada penelitian ini dilakukan analisa data menggunakan metode data mining dengan algoritma C4.5. Kemudian dibuatkan kesimpulan

berdasarkan analisa dan implementasi data bahwa pembuatan model menggunakan algoritma C4.5 menghasilkan sebuah aplikasi menggunakan bahasa pemrograman *Android*. Penerapan data baru menggunakan aplikasi tersebut menghasilkan data yang sesuai dengan prediksi, sehingga program tersebut dapat digunakan untuk penentuan kelayakan pemberian kredit koperasi.

2. Pratiwi dan Herliana (2015) dalam penelitian berjudul "Analisis dan Desain Sistem Informasi Simpan Pinjam Pada Koperasi Sejahtera Bersama Bandung" menjelaskan bahwa pengolahan data yang masih manual menyebabkan perlu waktu lama dalam etiap transaksinya. Maka penelitian ini dilakukan pengembangan sistem informasi menggunakan metodologi *waterfall*, diagram HIPO, dan tiga diagram UML yang terdiri dari *usecase* diagram, *activity* diagram, dan *class* diagram. Perancangan *database* menggunakan Microsoft Access 2007 dan bahasa pemrograman Microsoft Visual Basic 6.0. Hasil dari sistem ini adalah sistem ini dapat mempersingkat waktu transasi sehingga operasional dalam koperasi bisa berjalan lebih efektif.
3. Danny (2015) melakukan penelitian dengan judul "Sistem Informasi Pinjaman Dana Tunai Berbasis *Web* dengan Menggunakan Database *MySQL*". Penelitan ini menggunakan tahap pengumpulan data dari studi literatur, observasi, *interview* dan tahap pembuatannya diantaranya yaitu analisis, *design*, *coding*, pengujian dan *maintenance*. Hasil dari penelitian ini bahwa untuk dapat membangun suatu sistem informasi yang handal diperlukan suatu perancangan sistem informasi yang akurat serta didukung oleh kemampuan sumber daya

manusia yang memadai. Untuk itu perlu dibangunnya suatu sistem informasi secara *real time* yang berbasis teknologi komputer.

4. Nani, dkk. (2015) melakukan sebuah penelitian dengan judul "Perancangan Aplikasi Koperasi Simpan Pinjam Berbasis Jaringan *Localhost* dengan Menggunakan *VB.Net*". Penelitian ini menggunakan pemodelan *Unified Modelling Language* (UML) untuk memodelkan aplikasi sebagai teknik analisis. Perancangan programnya menggunakan *VB.Net, database SQL Server*, dan *Crystal Report* sebagai aplikasi untuk pembuatan laporannya. Aplikasi yang dihasilkan adalah aplikasi pengolahan data koperasi simpan pinjam berbasis jaringan *localhost*. Dari keseluruhan penelitian dapat disimpulkan bahwa aplikasi koperasi simpan pinjam berbasis jaringan *localhost* dapat membantu sistem kerja yang cepat dan tepat dalam peng-*input*-an data serta membantut pengolahan data.
5. Setiawan dan Rinawati (2015) dalam penelitian berjudul "Implementasi Aplikasi Peminjaman dan Pembayaran Angsuran pada BPR Kabupaten Bandung" menjelaskan penelitian ini bertujuan untuk merancang sebuah sistem yang dapat menangani sistem pengelolaan data pinjaman dan pembayaran angsuran nasabah dengan berbasis komputer. Penelitian ini menggunakan metode pengembangan *waterfall*. Analisis kebutuhan sistem diimplementasikan dengan program *Delphi 7* dan *Microsoft Access* sebagai pengolah data peminjaman dan pembayaran angsuran. Hasil dari sistem baru ini adalah dapat mengatasi permasalahan pada sistem yang lama. Sehingga menghasilkan sistem yang dapat mempermudah BPR Kabupaten Bandung dalam melakukan pengelolaan data angsuran.

2.2. Teori Pendukung

Teori pendukung adalah pembahasan yang digunakan untuk mendapatkan logika model dari suatu sistem yang menggunakan simbol-simbol, lambang-lambang, diagram-diagram yang menunjukkan secara tepat arti dan fungsinya.

2.2.1. Struktur Navigasi

Menurut Zamaludin, dkk (2016: 21) "Struktur navigasi adalah navigasi yang ada pada situs *web* atau aplikasi *web* menunjukan sesuatu yang penting dan menjadi kunci usabilitas aplikasi". Sebelum menyusun sebuah aplikasi harus menentukan terlebih dahulu alur apa yang akan digunakan. Bentuk dasar dari struktur navigasi yang biasa digunakan dalam proses pembuatan aplikasi, yaitu:

1. Struktur Navigasi *Linear*

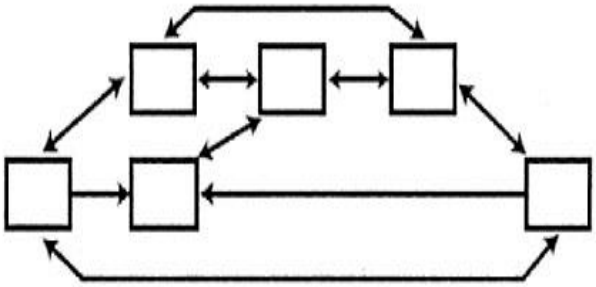
   Struktur navigasi *linear* (terurut) maka pengguna akan melakukan navigasi secara berurutan, dari *frame* atau *byte* informasi yang satu ke yang lainnya.

Sumber: Zamaludin, dkk (2016: 21)

Gambar II.2.
Struktur Navigasi *Linear*

2. Struktur Navigasi *Non Linear*

   Struktur navigasi *non linear* (tidak terurut) maka pengguna akan melakukan navigasi dengan bebas melalui isi proyek dengan tidak terikat dengan jalur yang sudah ditentukan sebelumnya.



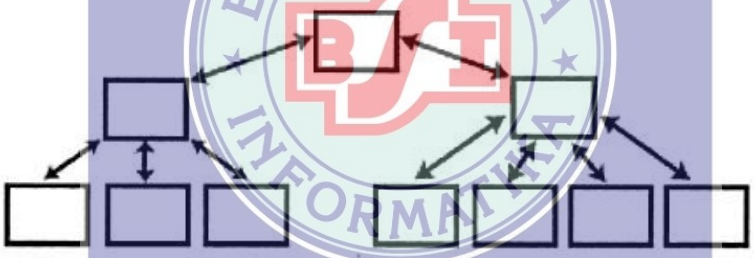
Sumber: Zamaludin, dkk (2016: 21)

Gambar II.3.

Struktur Navigasi *Non Linear*

3. Struktur Navigasi *Hierarchy*

Struktur dasar ini disebut juga struktur "*linieari* dengan percabangan" karena pengguna melakukan navigasi disepanjang cabang pohon struktur yang terbentuk oleh logika isi.



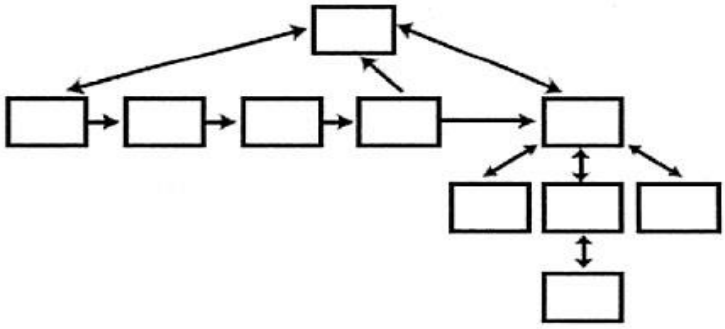
Sumber: Zamaludin, dkk (2016: 21)

Gambar II.4.

Struktur Navigasi *Hierarchy*

4. Struktur Navigasi *Composite*

Struktur navigasi *composite* (campuran) maka pengguna akan melakukan navigasi dengan bebas (secara *non-linear*), tetapi terkadang dibatasi presentasi *linear* film atau informasi penting pada data yang paling terorganisasi secara logis pada suatu hierarki.



Sumber: Zamaludin, dkk (2016: 22)

Gambar II.5.

Struktur Navigasi *Composite*

2.2.2. Enterprise Relationship Diagram

1. *Entity Relationship Diagram*

   Yanto (2016: 32) menjelaskan bahwa "ERD adalah suatu diagram untuk menggambarkan desain konseptual dari model konseptual suatu basis data relasional". Menurut Sukamto dan Shalahuddin (2013:50) "*Entity Relationship Diagram* (ERD) digunakan untuk pemodelan basis data relasional". Sehingga jika penyimpanan basis data menggunakan OODBMS maka perancangan basis data tidak perlu menggunakan ERD. *Entity Relationship Diagram* memiliki komponen sebagai berikut:

   a. Entitas (*Entity*)

      Entitas adalah suatu objek di dunia nyata yang dapat dibedakan dengan objek lainnya. Objek dapat berupa orang, benda ataupun hal lainnya. Entitas dapat digambarkan sebagai persegi panjang. Dari jenisnya entitas terbagi 2 (dua) yaitu:

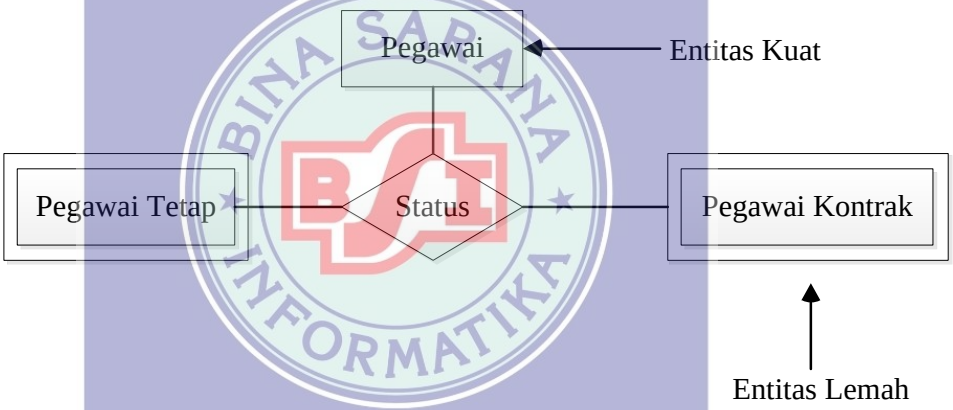
      1) Entitas Kuat (*Strong Entity*)

         Entitas kuat adalah entitas yang dapat berdiri sendiri tidak bergantung pada entitas lainnya, entitas kuat memiliki *atribut key* dan entitas kuat

digambarkan sebagai kotak persegi panjang bergaris tunggal. Contoh entitas kuat adalah pegawai.

2) Entitas Lemah (*Weak Entity*)

Entitas lemah adalah entitas yang tidak dapat berdiri sendiri. Entitas lemah merupakan hasil dari pembentukan entitas kuat, entitas lemah tidak memiliki *atribut key* dan entitas lemah digambarkan sebagai kotak 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.



Sumber: Yanto (2016: 34)

Gambar II.6.

Jenis Entitas

b. Atribut (*Atribute*)

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

1) Atribut Sederhana (*Simple Attribute*)

Atribut sederhana adalah atribut yang nilainya tidak dapat dibagi lagi menjadi banyak atribut yang lebih kecil. Contoh atribut sederhana adalah harga.

Sumber: Yanto (2016: 34)

Gambar II.7.

Atribut Sederhana

2) Atribut Komposit (*Composite Attribute*)

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. Contoh atribut komposit adalah alamat.

Sumber: Yanto (2016: 35)

Gambar II.8.

Atribut Komposit

3) Atribut Benilai Tunggal (*Single Values Attribute*)

Atribut bernilai tunggal adalah jenis atribut yang nilainya hanya satu dari suatu entitas. Contoh atribut bernilai tunggal adalah tanggal lahir dari entitas mahasiswa. Bisa dipastikan bahwa satu mahasiswa pasti mempunyai satu tanggal lahir.

Sumber: Yanto (2016: 35)

Gambar II.9.

Atribut Bernilai Tunggal

4) Atribut Bernilai Banyak (*Multivalues Attribute*)

Atribut bernilai banyak adalah jenis atribut yang nilainya lebih dari satu dalam suatu entitas tertentu. Contoh atribut bernilai banyak adalah hobi.

Sumber: Yanto (2016: 36)

Gambar II.10.

Atribut Bernilai Banyak

5) Atribut Turunan (*Derived 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. Atribut masa_bakti akan muncul dengan bantuan *query*.

Sumber: Yanto (2016: 36)

Gambar II.11.

Atribut Turunan

6) Atribut Identitas (*Key Attribute*)

Atribut identitas adalah atribut yang dijadikan sebagai kunci pada suatu entitas. Sifat atribut identitas unik, tidak ada yang menyamai. Contoh atribut identitas adalah NIM.

Sumber: Yanto (2016: 38)

Gambar II.12.
Atribut Identitas

c. Relasi (*Relationship*)

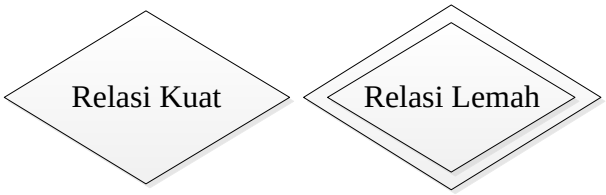
Relasi adalah hubungan alamiah yang terjadi antara satu atau lebih entitas. Gambar belah ketupat merupakan perlambangan relasi antar entitas atau sering disebut kerelasian. Ada 2 macam penggambaran relasi, yaitu:

1) Relasi Kuat

Relasi kuat adalah relasi yang berfungsi untuk menghubungkan antar entitas kuat.

2) Relasi Lemah

Relasi lemah adalah relasi yang berfungsi untuk menghubungkan entitas kuat dengan entitas lemah.



Sumber: Yanto (2016: 38)

Gambar II.13.
Tipe Relasi

d. Kardinalitas Relasi

Angka yang menunjukkan banyaknya kemunculan suatu obyek terkait dengan kemunculan obyek lain pada suatu relasi. Kardinalitas dibagi menjadi 3 (tiga) bagian yaitu:

1) Kardinalitas *One to One*

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

Sumber: Yanto (2016: 41)

Gambar II.14.
Kardinalitas *One to One*

2) Kardinalitas *One to Many*

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

Sumber: Yanto (2016: 41)

Gambar II.15.
Kardinalitas *One to Many*

3) Kardinalitas *Many to Many*

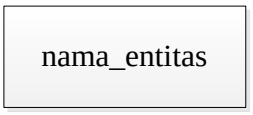
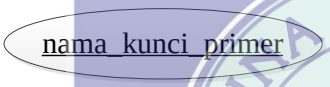
Kardinalitas *many to many* terjadi jika banyak entitas X berelasi dengan banyak entitas Y, ataupun sebaliknya. Sebagai contoh banyak mahasiswa belajar banyak matakuliah.

Sumber: Yanto (2016: 41)

Gambar II.16.
Kardinalitas *Many to Many*

Tabel II.1.

Simbol ERD

Simbol	Deskripsi
Entitas / *Entity* 	Entitas merupakan data inti yang akan disimpan; bakal tabel pada basis data dan harus disimpan datanya agar dapat diakses oleh aplikasi komputer; penamaan entitas biasanya lebih ke kata benda dan belum merupakan nama tabel
Atribut 	*Field* atau kolom data yang butuh disimpan dalam suatu entitas
Atribut kunci primer 	*Field* atau kolom data yang butuh disimpan dalam suatu entitas dan digunakan sebagai kunci akses *record* yang diinginkan; biasanya berupa id; kunci primer dapat lebih dari satu kolom, asalkan kombinasi dari beberapa kolom tersebut dapat bersifat unik (berbeda tanpa ada yang sama)
Atribut multinilai / *multivaue* 	*Field* atau kolom data yang butuh disimpan dalam suatu entitas yang dapat memiliki nilai lebih dari satu
Relasi 	Relasi yang menghubungkan antar entitas; biasanya diawali dengan kata kerja
Asosiasi / *association* 	Penghubung antara relasi dan entitas di mana di kedua ujungnya memiliki *multiplicity* kemungkinan jumlah pemakaian.

Sumber : Sukamto dan Shalahudin (2016:50)

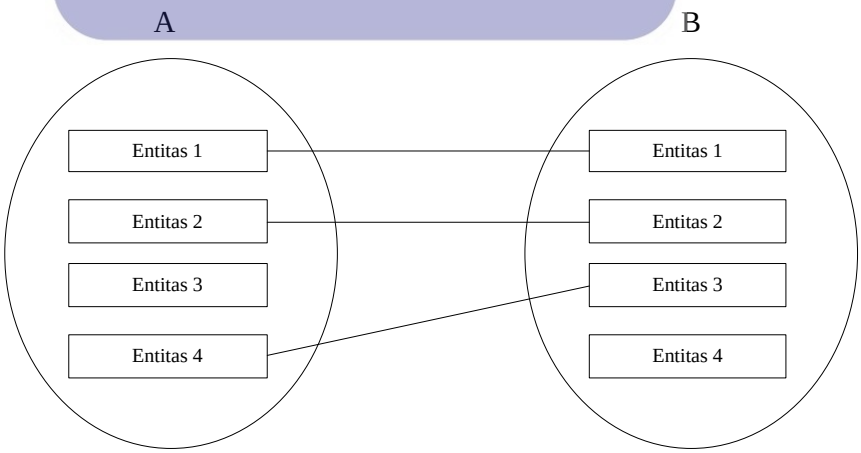
2. *Logical Record Structure*

Menurut Simarmata dan Paryudi dalam Fridayanthie dan Mahdiati (2016: 132) "*Logical Record Structure* adalah representasi dari struktur *record-record* pada tabel yang terbentuk dari hasil relasi antar himpunan entitas".

Zamaludin, dkk (2016: 22) menerangkan bahwa "LRS merupakan kepanjangan dari *Logical Record Structure* merupakan hasil dari pemodelan *Entity Relationship* (ER) beserta atributnya sehingga bisa terlihat hubungan-hubungan antara entitas".

Logical Record Structure dibentuk dengan nomor dari tipe record. Beberapa tipe *record* digambarkan oleh kotak empat persegi panjang dan dengan nama yang unik. Penggambaran LRS mulai dengan menggunakan model yang dimengerti. Dua metode yang dapat digunakan, dimulai dengan hubungan kedua model yang dapat dikonversikan ke LRS. Metode yang lain dimulai dengan Er-diagram dan langsung dikonversikan ke LRS. Untuk menentukan kardinalitas, jumlah tabel dan *Foreign Key* sebagai berikut:

a. *One to one* : Relasi 1:1 akan membentuk 2 tabel

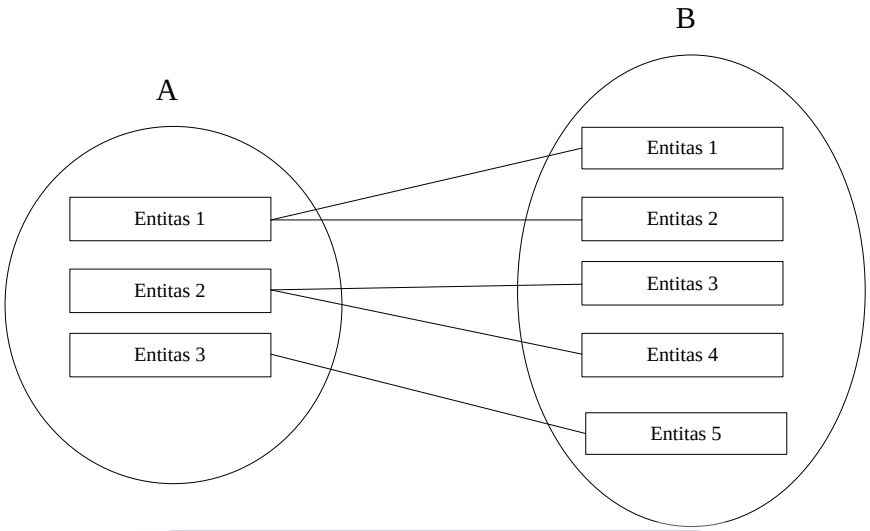


Sumber : Fathansyah (2018: 79)

Gambar II.17.

Relasi *One to One*

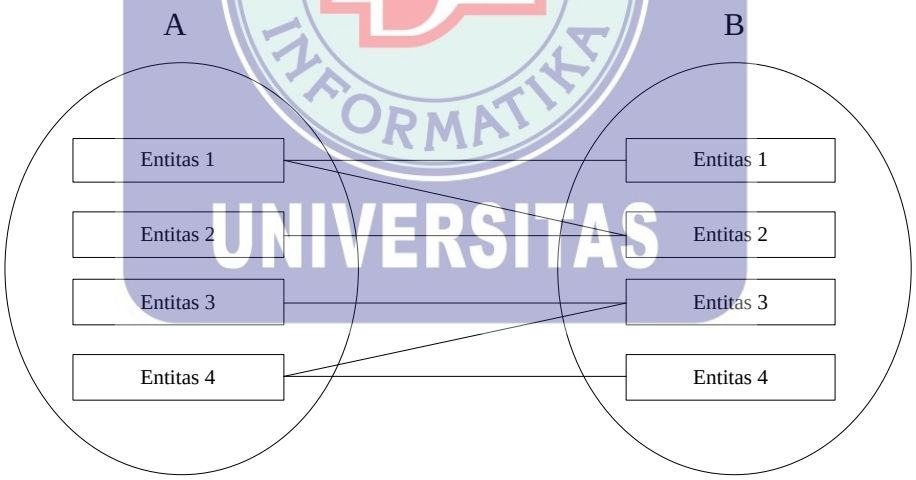
b. *One to many* : Relasi 1:M akan membentuk 2 tabel



Sumber : Fathansyah (2018: 80)

Gambar II.18.
Relasi *One to Many*

c. *Many to many* : Relasi M:N akan membentuk 3 tabel



Sumber : Fathansyah (2018: 81)

Gambar II.19.
Relasi *Many to Many*

2.2.3. Implementasi dan Pengujian *Web*

Pengujian fokus pada perangkat lunak dari segi lojic dang fungsional dan memastikan bahwa semua bagian sudah diuji. Hal ini dilakukan untuk

meminimalisir kesalahan (*error*) dan memastikan keluaran yang dihasilkan sesuai dengan yang diinginkan.

Menurut Sukamto dan Shalahudin (2016: 275) *Blackbox Testing* yaitu "Menguji perangkat lunak dari segi spesifikasi fungsional tanpa menguji desain dan kode program. Pengujian dimaksudkan untuk mengetahui apakah fungsi-fungsi masukan dan keluaran dari perangkat lunak sesuai dengan spesifikasi yang dibutuhkan".

Pengujian kotak hitam dilakukan dengan membuat kasus uji yang bersifat mencoba semua fungsi dengan perangkat lunak apakah sesuai dengan spesifikasi yang dibutuhkan. Kasus uji yang dibuat untuk melakukan pengujian kotak hitam harus dibuat dengan kasus benar dan kasus salah. Misalkan untuk kasus proses login maka kasus uji yang dibuat adalah:

1. Jika *user* memasukkan nama pemakai (*username*) dan kata sandi (*password*) yang benar.
2. Jika *user* memasukkan nama pemakai (*username*) dan kata sandi (*password*) yang salah, atau sebaliknya, atau keduanya salah.

Tabel II.2.

Contoh Tabel *Blackbox Testing*

No	Skenario Pengujian	Test Case	Hal yang Diinginkan	Hasil Pengujian	Kesimpulan
1	*Username* dan *Password* tidak diisi kemudian klik tombol *login*.	*Username :* Kosong *Password :* Kosong	Sistem akan menolak akses *user* dan menampilkan "Maaf semua *field* harus diisi".	Sesuai Harapan	Valid
2	Mengetikan *Username* diisi dan *Password* tidak diisi atau kosong kemudian klik tombol *login*.	*Username :* Ichal *Password :* Kosong	Sistem akan menolak akses *user* dan menampilkan "Maaf semua *field* harus diisi".	Sesuai Harapan	Valid
3	*Username tidak* diisi dan *Password* diisi kemudian klik tombol *login*.	*Username :* Kosong *Password :* Ichal	Sistem akan menolak akses *user* dan menampilkan "Maaf semua *field* harus diisi".	Sesuai Harapan	Valid
4	Mengetikan salah satu kondisi salah pada *Username* atau *Password* kemudian klik tombol *login*.	*Username :* Ichal (Benar) *Password :* Ichal123 (Salah)	Sistem akan menolak akses *user* dan menampilkan "Maaf *Username* dan *Password* tidak valid".	Sesuai Harapan	Valid
5	Mengetikan *Username* dan *Password* dengan data yang benar kemudian klik tombol *login*.	*Username :* Ichal *Password :* Ichal94	Sistem menerima akses *login* dan kemudian langsung menampilkan menu utama.	Sesuai Harapan	Valid

Sumber : Zamaludin, dkk (2016: 25)