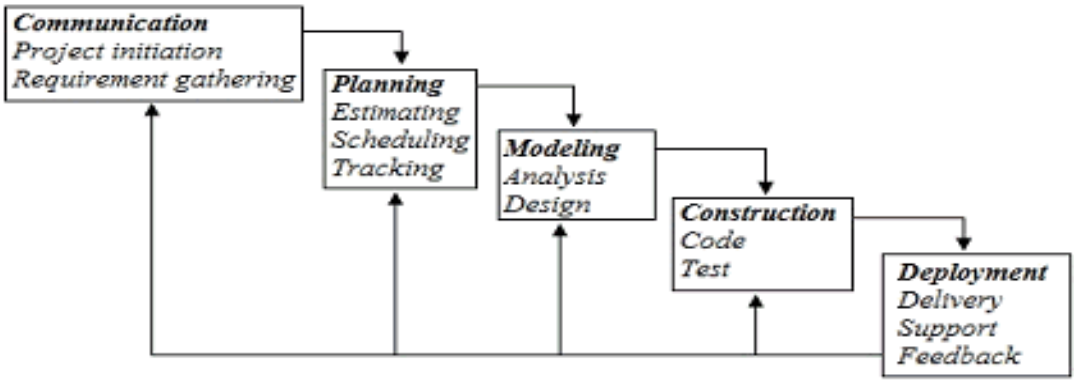
BAB II

LANDASAN TEORI

2.1 Tinjauan Pustaka

A. Konsep Dasar Model Pengembangan Sistem

Menurut Pressman (2006:79) "model *waterfall* adalah proses pengembangan *software* sekuensial, dimana kemajuan dipandang sebagai terus mengalir ke bawah (seperti air terjun) melalui tahapan konsepsi, inisiasi, analisis, desain, konstruksi, pengujian dan pemeliharaan."



Sumber : Pressman (2006:79)

Gambar II.1

***Waterfall* Pressman**

1. *Communication*

   Langkah ini merupakan analisis terhadap kebutuhan *software*, dan tahap untuk mengadakan pengumpulan data dengan melakukan pertemuan dengan *customer*, maupun mengumpulkan data-data tambahan baik yang ada di jurnal, artikel, maupun dari internet.

Pada tahap ini penulis melakukan pertemuan dengan bagian *operator* dan *editor* untuk melihat cara kerja, pengerjaan dan dokumen yang dihasilkan dan diperlukan.

2. *Planning*

Proses planning merupakan lanjutan dari proses *communication (analysis requirement).* Tahapan ini akan menghasilkan dokumen *user requirement* atau bisa dikatakan sebagai data yang berhubungan dengan keinginan *user* dalam pembuatan *software,* termasuk rencana yang akan dilakukan.

Setelah penulis melakukan pertemuan dengan pihak operator dan editor penulis membuatkan rangkuman infromasi *field-field* berdasarkan kegiatan yang dilakukan oleh pihak *operator* dan *editor* yang nantinya akan digunakan untuk perancangan *database.*

3. *Modelling*

Proses *modeling* ini akan menerjemahkan syarat kebutuhan ke sebuah perancangan *software* yang dapat diperkirakan sebelum dibuat *coding.*

Proses ini berfokus pada rancangan struktur data, arsitektur *software,* representasi *interface,* dan detail algoritma procedural. Tahapan ini akan Menghasilkan dokumen yang disebut *software requirement.*

Pada tahap ini penulis membuatkan konsep tampilan yang nantinya akan dibangun. Berikut desain *database* yang akan digunakan paada aplikasi web media *monitoring.*

4. *Construction*

Construction merupakan proses membuat kode. *Coding* atau pengkodean merupakan penerjemahan desain dalam bahasa yang bisa dikenali oleh komputer. *Programmer* akan menerjemahkan transaksi yang diminta oleh *user*.

Tahapan inilah yang merupakan tahapan secara nyata dalam mengerjakan suatu *software*, artinya penggunaan komputer akan dimaksimalkan dalam tahapan ini.

Setelah pengkodean selesai maka akan dilakukan testing terhadap sistem yang telah dibuat tadi. Tujuan *testing* adalah menemukan kesalahan-kesalahan terhadap sistem untuk kemudian bisa diperbaiki.

5. *Deployment*

Tahap ini bisa dikatakan final dalam pembuatan sebuah *software* atau sistem. Setelah melakukan analisis, desain dan pengkodean maka sistem yang sudah jadi akan digunakan oleh *user*. Kemudian *software* yang telah dibuat harus dilakukan pemeliharaan secara berkala.

B. Konsep Dasar Pemrograman

1. Hypertext Prepocessor (PHP)

Salah satu keunggulan *Hypertext Prepocessor (PHP)* dibanding bahasa pemograman lainnya adalah *PHP* dapat diperoleh secara gratis, meskipun bukan berarti karena gratis kemampuannya menjadi pas-pasan. *PHP* sangat *powerfull*. Terbukti dengan banyaknya website yang dibangun menggunakan PHP.

Menurut Musyawarah (2005:72) *PHP* adalah "instruksi atau perintah pemrograman berbasis web yang biasa disisipkan dalam *kode Hypertext Markup Language (HTML)* sebagai *script* pendukung yang ada di lingkungan *server".* Artinya, semua sintaks *PHP* yang diberikan akan sepenuhnya dijalankan pada *server*, sedangkan yang dikirimkan ke *browser* hanya hasilnya saja.

2. My Structured Query Language (MySQL)

Menurut Musyawarah (2005:03) basisdata atau *database* adalah "sekumpulan data yang berisi informasi mengenai satu atau beberapa *object".* Untuk mengolah *database* diperlukan suatu perangkat lunak yang disebut *Database Management System (DBMS). DBMS* merupakan suatu perangkat lunak yang memungkinkan pengguna untuk membuat. Memelihara, mengontrol dan mengakses *database* secara praktis dan efisien.

Dengan *DBMS*, user akan lebih mudah mengontrol dan memanipulasi data yang ada. RDMS atau *Relationship Database Management System* merupakan salah

satu jenis *DBMS* yang mendukung adanya *relationship* atau hubungan antar tabel. *MySQL* merupakan suatu *DBMS* yang sudah mendukung adanya hubungan antar tabel.

3. Object Oriented Programming (OOP)

Ada dua metode dalam membuat kontruksi aplikasi *website*, yaitu pemrograman terstruktur atau *procedural* dan pemrograman berorientasi objek. Pemgoraman berorientasi objek *(OOP)* adalah metode pemrograman dimana *web deveploper* membuat dan mengelompokkan kode-kode yang berkaitan menjadi suatu objek. Sehingga setiap objek dapat memiliki data dan fungsi sendiri. Dan fungsi tersebut dapat digunakan dengan memanggil objek yang bersangkutan terlebih dahulu.

Salah ssatu kemungkinan *OOP* dibandingkan teknik pemrograman terstruktur adalah *OOP* mungkin *web deveploper* untuk membuat modul yang tidak perlu berubah ketika suatu objek harus ditambahkan. Bahkan *web deveploper* dapat membuat suatu objek baru yang mewarisi beberapa fitur dari objek yang sudah ada, hal ini membuat aplikasi yang berorientasi objek lebih mudah dimodifikasi atau diperbaiki dan dikembangkan dibandingkan pemrograman terstruktur.

4. Model View Controller (MVC)

Menurut Wardana (2013:3) *Framework* adalah "kumpulan perintah atau fungsi dasar yang membentuk aturan-aturan tertentu dan saling berinteraksi satu sama lain sehingga dalam pembuatan aplikasi *website,* kita harus mengikuti aturan dari *framework* tersebut".

Dimana *framework* sendiri menggunakan *konsep MVC (Model View controller)* yang menerapkan teknik pemrograman berbasis objek. MVC sendiri yakni konsep yang memisahkan antara data *(Model),* tampilan *(View).* Dan bagaimana cara memprosesnya *(Controller).*

MVC memisahkan pengembangan aplikasi berdasarkan komponen utama yang membangun sebuah aplikasi seperti manipulasi data, antarmuka pengguna dan bagian yang menjadi kontrol dalam sebuah aplikasi web.

5. Peralatan Pendukung Sistem

a. UML *(Unified Modelling Language)*

Menurut Munawar (2005:17) " *Unifed Modelling Language (UML)* adalah salah satu alat bantu yang sangat handal di dunia pengembangan sistem yang berorientasi objek". *UML* merupakan kesatuan dari bahasa pemodelan yang dikembangkan oleh Booch, *Object Modelling Technique (OMT)* dari *object Oriented Software Engineering (OOASE).* Metode Booch dari Grady Booch sangat terkenal dengan nama metode *Design Object Oriented.* Metode ini dijadwalkan proses analisis dan design ke empat tahapan iteraktif, yaitu: identifikasi kelas-kelas dan objek-objek, identifikasi sematik dari hubungannya dengan objek dan kelas tersebut, perincian *interface* dan implementasi.

1. *Use Case Diagram*

" *Use Case* diagram adalah deskripsi fungsi dari sebuah menggambarkan fungsionalitas yang diharapkan dari sebuah sistem" (Munawar, 2005:63). Dalam pembicaraan *use case,* pengguna biasanya disebut dengan *actor, actor* adalah sebuah peran yang bisa dimainkan oleh pengguna data interaksinya dengan sistem, diagram use case menunjukkan 3 aspek dari sistem yaitu *actor, use case* dan *system* atau sub *system boundary. Actor* mewakili peran orang, sistem yang lain atau alat ketikas berkomunikasi dengan *use case.*

2. *Activity Diagram*

"Acivity Diagram adalah teknik untuk mendeskripsikan logika prosedural, proses bisnis dan aliran kerja dalam banyak kasus" (Munawar, 2005:109). *Activity diagram* mempunyai peran seperti halnya *flowchart,* akan tetapi perbedaannya dengan *flowchart* adalah *activity* diagram mendukung perilaku pararel, sedangkan *flowchart* tidak bisa. *Activity* diagram menunjukkan tahapan, pengambilan keputusan dan percabangan, diagram ini sangat berguna untuk menunjukkan *operation* sebuah sistem objek dan proses bisnis. Kelebihan *activity* diagram dibanding *flowchart* adalah kemampuannya dalam aktifitas pararel.

3. *Sequence Diagram*

digunakan untuk menggambarkan perilaku pada sebuah *scenario.* Diagram ini menunjukkan sejumlah contoh objek dan message (pesan) yang diletakkan diantara objek-objek ini di dalam *use case.* Komponen utama *sequence* diagram terdiri atas

objek yang dituliskan dengan kotak segiempat bernama *"Message* diwakili oleh garis dengan tanda panah dan waktu yang ditunjukkan dengan progress *vertical"* (Munawar, 2005:87).

4. *Component Diagram*

Component diagram mempresentasikan dunia rill item yaitu *component software,* component bisa diakses melalui *interfacenya* yaitu koleksi operasi-operasi. Relasi antara *component* dan *interfacenya* disebut *realization,* suatu component bisa mengakses *service-service* yang ada di component lain dengan cara *import interface.* Sedangkan component yang menyediakan service menggunakan *export interface.* "hal ini penting dari *component* adalah component mewakili potongan-potongan yang independen yang bisa dipesan dan diperbaharui sewaktu-waktu" (Munawar, 2005:124).

5. *Deployment Diagram*

Deployment diagram menunjukkan tata letak sebuah sistem secara fisik, menampakkan bagian-bagian *software* yang berjalan pada bagian-bagian *hardware*. *Deployment* diagram menyediakan gambaran bagaimana sistem secara fisik akan terlihat, sistem terdiri dari *node-node* dimana setiap *node* diwakili untuk sebuah kubus (Munawar, 2005:125).

C. ERD *(Entity Relationship Diagram)*

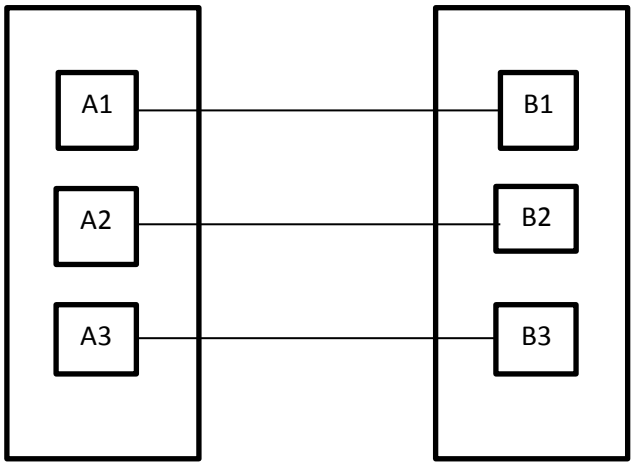
Menurut Simarmata dan Prayudi (2006:67). "*Entity Relationship Diagram* adalah alat pemodelan data utama yang akan membantu mengorganisasi data dalam suatu proyek ke dalam entitas-entitas dan menentukan hubungan antar entitas. Proses memungkinkan struktur basisdata yang baik sehingga data dapat disimpan dan diambil secara efisien."

ERD menggunakan sejumlah notasi dan symbol untuk menggambarkan struktur dan hubungan antar data beberapa symbol yang digunakan dalam ERD pada gambar berikut :

Menurut Fatansyah dan Ariyanto (2008:137) “kardinalitas ialah spesifikasi dari sejumlah peristiwa dari satu obyek yang dapat dihubungkan ke sejeumlah persitiwa dari obyek lain, kardinalitas mendefinisikan jumlah maksimum dari hubungan obyek yang ikut serta di dalam suatu hubungan”. Untuk suatu himpunan relasi biner R antara himpunan A dan entitas B, kardinalitas yang terjadi harus salah satu dari bentuk berikut:

1. *One-to-one*

   Sebuah entitas pada A berhubungan dengan paling banyak satu entitas pada B dan sebuah entitas pada B berhubungan dengan paling banyak satu entitas pada A.



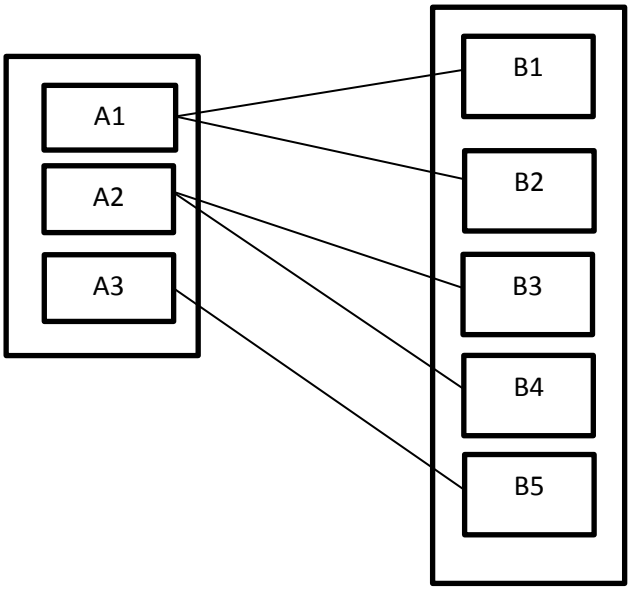
Gambar II.2

Hubungan *One to one*

2. *One-to-many*

   Sebuah entitas pada A hubungan dengan nol atau lebih entitas pada B.

   Sebuah entitas pada B dapat dihubungkan dengan paling banyak satu entitas pada A.

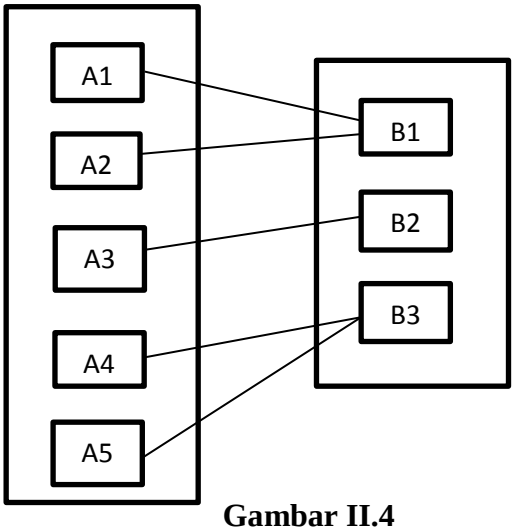


Gambar II.3

Hubungan *One to Many*

3. *Manty-to-one*

Sebuah entitas pada A hubungan dengan paling banyak satu entitas pada B. sebuah entitas pada B dapat dihubungkan dengan nol atau lebih entitas pada A.

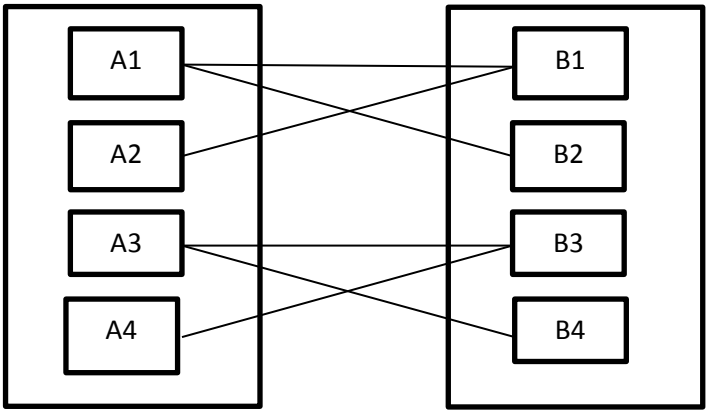


Gambar II.4

Hubungan *Many to one*

4. *Many-to-many*

Sebuah entitas pada A berhubungan dengan nol atau lebih entitas pada B dan sebuah entitas pada B dapat dihubungkan nol atau lebih entitas pada A.



Gambar II.5

Hubungan *Many to many*

D. LRS *(Logical Record Structure)*

Menurut Friyadie (2007:13) *LRS (Logical Record Structure)* merupakan "hasil pemodelan *Entity Relationship* (ER) beserta atributnya sehingga bisa terlihat hubungan-hubungan entitas".

Menurut Friyadie (2017:14) dalam pembuatan *LRS* terdapat tiga hal yang dapat mempengaruhi, yaitu :

1. Jika tingkat hubungan *(cardinality)* satu pada satu *(one to one)* maka digabungkan dengan entitas yang memiliki atribut yang lebih sedikit.
2. Jika tingkat hubungan *(cardinality)* satu pada banyak (*one to many),* maka hubungan relasi atau digabungkan dengan entitas yang tingkat hubungannya banyak.
3. Jika tingkat hubungan *(cardinality)* banyak pada banyak *(many to many),*maka hubungan relasi tidak akan digabungkan dengan entitas manapun, melainkan menjadi sebuah LRS.

2.2 Penelitian Terkait

Sistem informasi pemesanan dan transaksi pembayaran akan lebih efektif dan efisien dengan komputerisasi yang tepat. Sistem informasi pemesanan dan transaksi pembayaran mampu memberikan kecepatan pelayanannya, pengolahan data yang akurat, sehingga data bisa langsung digunakan dan dilaporkan kepada atasan.

Sistem informasi pengolahan data pemesanan dan transaksi pembayaran ini sebelumnya sudah banyak dibuat, tetapi tempat dan program aplikasinya yang digunakan berbeda-beda. Adapun sistem informasi yang berkaitan dengan penjualan yang pernah dibuat adalah sebagai berikut:

Menurut Rejeki, dkk (2011:150) mengemukakan bahwa, mekanisme sistem penjualan pada distro smith sekarang masih menggunakan sistem *konvensional*, dimana konsumen harus dating langsung ke distro. Dengan adanya Aplikasi *Ecommerce* pada distro smith semarang diharapkan dapat memberikan kemudahan kepada masyarakat untuk melakukan pembelian produk tanpa harus dating ke tempatnya, serta memperluas pemasaran dan meningkatkan *costumer loyality*.

Menurut Pailin (2012:67) Toko Ribo Jaya yang merupakan usaha dagang jual-beli pakaian, agar mampu menyediakan informasi yang tepat dan akurat dalam melakukan transaksinya. Sistem yang berjalan di toko Ribo Jaya saat ini masih dilakukan secara manual seperti pencatatan transaksi pembelian, pencacatan transaksi penjualan, pencarian barang, perhitungan stok, dan pembuatan laporan-laporan yang mana memerlukan waktu yang cukup lama dan resiko kesalahan manusia *(human error)* yang cukup besar, berdasarkan penelitian dan pengujian program yang dilakukan, terjadi efisiensi/pengurangan waktu 63% dari waktu rata-rata keseluruhan 10 jenis barang secara manual dimana waktu yang dibutuhkan 22.27 menit dengan menggunakan program menjadi 8.09 menit dan mengurangi tingkat kesalahan karena perhitungan telah dilakukan secara otomatis oleh program. Dari keseluruhan hasil penelitian, untuk penghematan waktu dan biaya operasional bentuk usaha dagang sebaiknya Toko Ribo Jaya menggunakan program sistem informasi dalam melakukan transkasi penjualannya.