BAB 2
TINJAUAN PUSTAKA

2.1 Tinjauan perusahaan

Tinjauan perusahaan terdiri dari profil singkat perusahaan, struktur organisasi dan fungsi perusahaan.

2.1. 1 Profil Singkat Perusahaan

Toko Heinzy Komputer merupakan toko milik pribadi yang bergerak dalam bidang penjualan peralatan komputer. Toko ini dirintis oleh pemiliknya saudara Hendi Satria sejak tahun 2010 hingga sekarang. Beralamat di areal pertokoan BTM (Bandung Trade Mall) Blok GF A1 No 7 yang terletak di jalan cicadas Bandung. Peralatan komputer yang dijual di toko Heinzy Komputer terdiri dari peralatan yang baru dan peralatan yang bekas pakai. Penjualan barang di toko ini dilakukan dengan cara penjual berhubungan langsung ke konsumen dan menggunakan prinsip B2C(*Bussiness to Customer*) yang dengan kata lain penjual memasarkan produknya langsung ke konsumen untuk selanjutnya digunakan untuk kepentingan konsumen tersebut.

2.1. 2 Logo Perusahaan

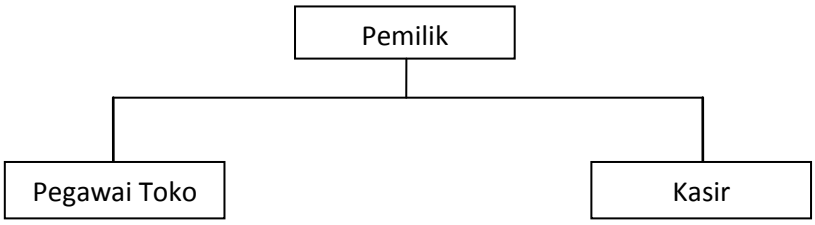
Berikut adalah logo perusahaan toko Heinzy Komputer akan ditunjukkan pada gambar 2.1 berikut :



Gambar 2. 1 Logo Perusahaan “Heinzy Komputer”

2.1. 3 Struktur Organisasi dan Fungsi Perusahaan

Dalam menjalankan kegiatan usahanya, toko Heinzy komputer mempunyai struktur organisasi dengan jabatan dan tugas masing-masing pada setiap bagian, sehingga kegiatan operasional dapat berjalan dengan baik dan lancar.



Gambar 2.2 Struktur Organisasi Heinzy Komputer

Secara umum tugas dan fungsi organisasi dari masing-masing bagian adalah sebagai berikut :

a. Pemilik

1. Memiliki wewenang dalam menentukan kebijakan toko.
2. Menetapkan target atau memberikan intruksi pada pegawai.
3. Mengatur pengadaan barang yang ada di toko.
4. Menyimpan semua data dan dokumen toko.

b. Pegawai toko

1. Melayani dan bertatap muka langsung dengan pelanggan.
2. Memelihara barang yang ada di toko.
3. Melakukan pengecekan stok barang dan melaporkan ke pemilik apabila stok barang kurang atau habis.
4. Menerima pengaduan retur.

c. Kasir

1. Mengatur keuangan toko
2. Menyimpan kumpulan nota penjualan.
3. Membuat laporan penjualan.

2.2 Landasan Teori

Landasan teori merupakan ilmu dan teori pendukung yang dapat dijadikan referensi atau pegangan dalam pembangunan aplikasi ini.

2.2. 1 Konsep Dasar Sistem Informasi

Sistem informasi dalam sebuah sistem meliputi pemasukan data (*input*) kemudian diolah melalui suatu model dalam pemrosesan data, dan hasil informasi akan ditangkap kembali sebagai suatu input dan seterusnya sehingga membentuk siklus informasi yang dapat diperoleh dari sistem informasi sebagai sistem khusus dalam organisasi untuk mengolah informasi tersebut

Sistem informasi adalah suatu sistem yang dibuat oleh manusia yang terdiri dari komponen-komponen dalam organisasi untuk mencapai suatu tujuan yaitu menyajikan informasi. Pengertian lain dari sistem informasi adalah sekumpulan prosedur organisasi yang pada saat dilaksanakan dan memberikan informasi bagi pengambil keputusan dan/atau untuk mengendalikan informasi. [4]

2.2. 2 Komponen Sistem Informasi

Komponen-komponen sistem informasi adalah sebagai berikut:

a. *Hardware* yaitu suatu perangkat keras dalam komputer yang kita bisa sentuh dan rasakan.
b. *Software* yaitu suatu perangkat lunak di dalam komputer yang berfungsi untuk mengoperasikan suatu aplikasi di dalam sistem komputer.
c. Data yaitu sekumpulan karakter yang diterima sebagai masukan (*input*) untuk sistem informasi dan disimpan serta diolah.
d. Prosedur yaitu suatu urutan pekerjaan tata usaha yang biasanya melibatkan beberapa orang dalam satu bagian atau lebih, dan disusun untuk menjamin adanya perlakuan yang seragam terhadap transaksi-transaksi perusahaan yang terjadi.
e. *User* yaitu orang yang terlibat dalam sistem informasi seperti operator, pemimpin sistem informasi, dan sebagainya.

2.2. 3 Tujuan Sistem Informasi

Sistem Informasi memiliki beberapa tujuan, yaitu:

1. Integrasi sistem
   a. Menghubungkan sistem individu/kelompok
   b. Pengkolektifan data dan penyambungan secara otomatis
   c. Peningkatan koordinasi dan pencapaian sinergi
2. Efisiensi pengelolaan
   a. Penggunaan basis data dalam upaya kesamaan pengadministrasian data
   b. Pengelolaan data berkaitan dengan karakteristik Informasi
   c. Penggunaan dan pengambilan Informasi
3. Dukungan keputusan untuk manajemen
   a. Melengkapi Informasi guna kebutuhan proses pengambilan kebutuhan
   b. Akuisisi Informasi eksternal melalui jaringan komunikasi.
   c. Ekstraksi dari Informasi internal yang terpadu

2.2. 4 Manfaat Sistem Informasi

Sistem Informasi memiliki beberapa manfaat, yaitu:

1. Menghemat tenaga kerja
2. Peningkatan efisiensi
3. Mempercepat proses
4. Perbaikan dokumenasi
5. Pencapaian standar
6. Perbaikan keputusan

2.2. 5 Elektronic Commerce

E-commerce (*Electronic Commerce)* atau perdagangan elektronikmerupakan suatu cara berbelanja atau berdagang secara *online* atau *direct selling* yang memanfaatkan fasilitas internet dimana terdapat *website* yang dapat menyediakan layanan "*get and deliver*" *e-commerce* akan merubah semua kegiatan marketing dan juga sekaligus.

Perdagangan elektronik melakukan hal yang mirip dengan perdagangan tradisional, tetapi ia memiliki kelebihan-kelebihan yang secara langsung dapat bermanfaat untuk meningkatkan pendapatan dan keuntungan perusahaan. Dengan fleksibilitasnya, perdagangan elektronik dapat memangkas biaya-biaya pemasaran dengan kemudahannya dan kecanggihannya dalam menyampaikan informasi tentang barang dan jasa langsung ke konsumen di mana pun mereka berada (tidak terbatas oleh jangkauan geografis perusahaan). [6]

2.2. 6 Jenis-Jenis E-commerce

E-commerce dapat dibagi menjadi dua jenis, yaitu *Business to Business* (B2B) dan *Business to Consumer* (B2C, *retail).* Kedua jenis *e-commerce* ini memiliki karakteristik yang berbeda, diantaranya adalah sebagai berikut : [6]

1. *Business to Business e-commerce* (B2B) memiliki karakteristik :
   a. *Trading partners* yang sudah saling mengetahui dan antara mereka sudah terjalin hubungan yang berlangsung cukup lama. Pertukaran informasi hanya berlangsung diantara mereka karena sudah sangat mengenal, maka pertukaran informasi tersebut dilakukan atas dasar kebutuhan dan kepercayaan.
   b. Pertukaran data dilakukan secara berulang-ulang dan berkala dengan format data yang telah disepakati. Jadi *service* yang digunakan antara kedua sistem tersebut sama dan menggunakan standar yang sama pula.
   c. Salah satu pelaku tidak harus menunggu partner mereka lainnya untuk mengirimkan data.
   d. Model yang umum digunakan adalah *peer-to-peer*, dimana *processing intelligence* dapat didistribusikan dikedua pelaku bisnis.
2. *Business to consumer e-commerce* (B2C) memiliki karakteristik :
   a. Terbuka untuk umum, dimana informasi disebarkan secara umum pula.
   b. *Service* yang dilakukan juga bersifat umum sehingga mekanismenya dapat digunakan oleh orang banyak. Sebagai contoh, karena sistem web sudah umum dikalangan masyarakat maka sistem yang digunakan adalah sistem web pula.

c. *Service* yang diberikan adalah berdasarkan permintaan. Konsumen berinisiatif sedangkan produsen harus siap memberikan respon terhadap inisiatif konsumen tersebut.

Sering dilakukan sistem pendekatan *client-server*, dimana konsumen di pihak *client* menggunakan sistem yang minimal (berbasis web) dan penyedia barang/jasa (*business procedure*) berada pada pihak *server*.

2.2. 7 Basis Data (*Database*)

Basis data terdiri atas 2 kata, yaitu Basis dan Data. Basis kurang lebih dapat diartikan sebagai markas atau gudang, tempat bersarang atau berkumpul.Sedangkan data adalah representasi fakta dunia nyata yang mewakili suatu objek seperti manusia (pegawai, siswa, pembeli, pelanggan), barang, hewan, peristiwa, konsep, keadaan, dan sebagainya yang direkam dalam bentuk angka, huruf, symbol, teks, gambar, bumi, atau kombinasinya. [5]

2.2. 8 Diagram Aliran Dokumen (Flowmap)

Bagan alir dokumen atau disebut juga bagan alir formulir merupakanbagan alir yang menunujukan arus dari laporan dan formulir termasuk tembusantembusannya.Bagan alir dokumen menggambarkan aliran dokumen dan informasiarus antar area pertanggung jawaban di dalam sebuah organisasi.Secara rincibagan alir ini menunjukan dari mana dokumen tersebut berasal, distribusinya,tujuan digunakannya dokumen tersebut dan lain-lain.Bagan alir ini bermanfaatuntuk menganalisis kecukupan prosedur pengawasan dalam sebuah sistem.

2.2. 9 ERD (Entity Relationship Diagram)

Entity Relationship Diagram adalah pemodelan data utama dan akan membantu mengorganisasikan data dalam suatu proyek ke dalam entitas-entitas dan menentukan hubungan antar entitas. ERD merupakan notasi grafis dalam pemodelan data konseptual yang mendeskripsikan hubungan antara penyimpanan.ERD digunakan untuk memodelkan struktur dan hubungan antar

data, karena hal ini relative kompleks. Dengan ERD kita dapat menguji model dengan mengabaikan proses yang harus dilakukan ERD menggunakan sejumlah notasi dan simbol untuk menggambarkan struktur dan hubungan antar data, pada dasarnya ada 3 macam simbol yang digunakan yaitu : [5]

1. *Entitas (entity)*

   Entitas adalah sesuatu yang memiliki keberadaan yang unik dan berbeda, walaupun tidak harus dalam bentuk fisik. Abstraksi, misalnya, biasanya dianggap juga sebagai suatu entitas. Dalam pengembangan sistem, entitas digunakansebagai model yang menggambarkan komunikasi dan pemrosesan internal seperti misalnya membedakan dokumen dengan pemrosesan pesanan.

2. Relasi (*Relationship*)

   Relasi adalah hubungan alamiah yang terjadi antara satu atau lebih entitas, misal proses pembayaran pegawai. Kardinalitas menentukan kejadian suatu entitas untuk satu kejadian pada entitas yang berhubungan.Misal, mahasiswa bisa mengambil banyak mata kuliah.

3. *Atribut (Attribute)*

   Entitas atau entity mempunyai elemen yang disebut atribut, dan berfungsi mendeskripsikan karakter entitas.

4. *Kardinalitas (Cardinality)*

Menyatakan jumlah anggota entitas yang terlibat didalam relasi yang terjadi.dalam hal ini relasi yang terjadi akan membentuk relasi hubungan (*relationship instance).*

1. Hubungan satu ke satu *(One to one relationship).*

   Yang berarti entitas pada himpunan entitas A berhubungan paling banyak dengan satu entitas B, dan begitu juga sebaliknya setiap entitas pada himpunan entitas B berhubungan paling banyak dengan satu entitas pada himpunan entitas A.

2. Hubungan satu ke banyak *(One to many relationship)*

   Yang berarti entitas pada himpunan entitas A berhubungan dengan banyak entitas pada satu entitas B, tetapi tidak sebaliknya setiap entitas pada

himpunan entitas B berhubungan paling banyak dengan satu entitas pada himpunan entitas A.

3. Hubungan banyak ke banyak *(Many to many relationship).*

   Yang berarti entitas pada himpunan entitas A berhubungan dengan banyak entitas pada satu himpunan entitas B dan begitu juga sebaliknya setiap entitas pada himpunan entitas B berhubungan dengan banyak himpunan entitas A.

2.2. 10 Object Oriented Program (OOP)

Object Oriented Programming (OOP) adalah suatu metode pemrograman yang berbasiskan pada objek, secara singkat pengertian dari OOP adalah koleksi objek yang saling berinteraksi dan saling memberikan informasi satu dengan yang lainnya. Semua data dan fungsi di dalam paradigma ini dibungkus dalam kelas-kelas atau objek-objek. Konsep dasar dari Pemrograman

Berorientasi Objek Pemrograman orientasi-objek menekankan konsep berikut :

1. *Object*

   Object adalah sebuah struktur yang menggabungkan data dan prosedur untuk bekerja bersama-sama.

2. *Class*

   Class adalah kumpulan atas definisi data dan fungsi-fungsi dalam suatu unit untuk suatu tujuan tertentu. Sebuah *class* merupakan dasar dari modularitas dan struktur dalam pemrograman berorientasi objek.

3. *Abstraction*

   Abstraction adalah kemampuan sebuah program untuk melewati aspek informasi yang diproses olehnya, yaitu kemampuan untuk memfokus pada inti. Setiap objek dalam sistem melayani sebagai model dari “pelaku” abstrak yang dapat melakukan kerja, laporan dan perubahan keadaannya, dan berkomunikasi dengan objek lainnya dalam sistem, tanpa mengungkapkan bagaimana kelebihan

   ini diterapkan. Proses, fungsi atau metode dapat juga dibuat abstrak, dan beberapa teknik digunakan untuk mengembangkan sebuah pengabstrakan.

4. *Encapsulation*

Encapsulation memastikan pengguna sebuah objek tidak dapat mengganti keadaan dalam dari sebuah objek dengan cara yang tidak layak; hanya metode dalam objek tersebut yang diberi ijin untuk mengakses keadaannya. Setiap objek mengakses *interface* yang menyebutkan bagaimana objek lainnya dapat berinteraksi dengannya. Objek lainnya tidak akan mengetahui dan tergantung kepada representasi dalam objek tersebut.

5. *Polymorphism*

Polymorphism adalah kemampuan dua buah objek yang berbeda untuk merespon pesan permintaan yang sama dalam suatu cara yang unik.

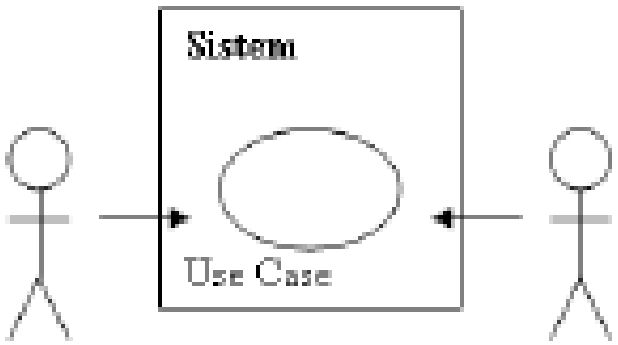
Ada beberapa diagram yang digunakan dalam Objek Oriented Program, diantaranya :

a) Use Case Diagram

Use case atau diagram *use case* merupakan pemodelan untuk kelakukan *(behavior)* sistem informasi yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. Secara kasar, *use case* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sebuah sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi itu. Syarat penamaan pada *use case* adalah nama didefinisikan sesimpel mungkin dan dapat dipahami.[9]

Ada dua hal utama pada *use case* yaitu pendefinisian apa yang disebut aktor dan *use case*.

1. Aktor merupakan orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat di luar sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang.
2. *Use case* merupakan fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor.



Gambar 2.3 Use Case Diagram

b) Class Diagram

Class adalah sebuah spesifikasi yang jika diinisiasi akan menghasilkan sebuah objek, kelas juga merupaka inti dari pengembangan dan desain berorientasi objek. *Class* menggambarkan keadaan atribut *(property)* pada suatu sistem, sekaligus menawarkan layanan untuk dapat memanipulasi keadaan atribut tersebut. *Class* memiliki empat area pokok, yaitu nama *class* yang disertai dengan *stereotype*, sebuah atribut, sebuah metode dan *multiplicity (dependency).* Sedangkan metode dan atribut memiliki salah satu sifat berikut :

1. *Private*, atribut yang hanya terlihat oleh kelas itu sendiri dan teman dari kelas tersebut.
2. *Protected*, atribut yang hanya terlihat oleh kelas itu sendiri, sub-kelasnya atau teman kelas tersebut.
3. *Public*, atribut yang dapat dilihat dari dalam dan dari luar paket yang mengandung kelas yang bersangkutan.
4. *Primer*, kelas yang berperan sebagai induk dari kelas lainnya.
5. *Esential*, sifat yang diturunkan dari kelas lainnya.

Selain memiliki sifat, atribut serta metode, *class* juga memiliki beberapa *multiplicity* atau sering dikenal dengan *dependency* untuk memperlihatkan fungsionalitas dari hubungan antar *class*, berikut jenis-jenis *multiplicity* yang menggambarkan hubungan antar *class*.



Gambar 2.4. *Multiplicity* Satu ke Satu



Gambar 2.5. *Multiplicity* Satu ke Banyak



Gambar 2.6. *Multiplicity* Satu ke Satu atau Banyak



Gambar 2.7. *Multiplicity* Satu atau Banyak ke Satu atau Banyak



Gambar 2.4. *Multiplicity* Banyak ke Banyak

Dari penjelasan tersebut maka dapat disimpulkan bahwa *class diagram* adalah sebuah diagram yang menggambarkan sebuah struktur dari deskripsi kelas, *package*, objek dan hubungan antar kelas seperti, *entertainment*, pewarisan, *association*, dan sebagainya.

c) Beh**aviour Diagram**

Behavior diagram dapat dikelompokkan menjadi tiga diagram, yaitu :

a. Statechart Diagram

Statechart diagram berfungsi untuk memodelkan prilaku dinamis satu kelas satu objek.

b. Activity Diagram

Activity diagram memodelkan alur kerja (*work flow*) sebuah proses bisnis dan urutan aktivitas dalam suatu proses.

c. Interaction Diagram

Iinteraction diagram dibagi menjadi dua model diagram yaitu :

1. *Sequence diagram* menjelaskan interaksi objek yang disusun dalam suatu urutan waktu. Diagram ini secara khusus bersosialisasi dengan *use case. Sequence diagram,* memperlihatkan tahap demi tahap apa yang seharusnya terjadi untuk menghasilkan sesuatu dalam *use case*

2. *Collaboration diagram* melihat pada interaksi dan hubungan terstruktur antar objek. Tipe diagram ini menekankan pada hubungan *(relationship)* antar objek, sedangkan *sequence diagram* menekankan pada urutan kejadian. Dalam *collaboration diagram* terdapat beberapa objek, *link*, dan *message*.

d) Implementation Diagram

Implementation diagram dibagi menjadi dua diagram, yaitu :

a) *Component Diagram* menggambarkan alokasi semua kelas dan objek ke dalam komponen-komponen dalam desain fisik sistem *software.* Diagram ini memperlihatkan pengaturan dan kebergantungan antara komponen-komponen *software*, seperti *source code, binary code,* dan komponen tereksekusi *(execute components)*

b) *Deployment Diagram* memperlihatkan pemetaan *software* kepada *hardware*. Dimana akan berjalan (di server/multitier, *standalone* atau lainnya), dan menggambarkan model koneksi dan kemampuan jaringan dan hal lainya yang bersifat fisik.

2.2. 11 Unified Modeling Language (UML)

Bahasa pemrograman berorientasi objek yang pertama dikembangkan dikenal dengan nama Simula-67 yang dikembangkan pada tahun 1967. Bahasa pemrograman ini urang berkembang dan dikembangkan lebih lanjut, namun

dengan kemunculannya telah memberikan sumbangan yang besar pada developer pengembang bahasa pemrograman berorientasi objek selanjutnya. Perkembangan aktif dari pemrograman berorientasi objek mulai menggeliat ketika berkembangnya bahasa pemrograman Smalltalk pada awal 1980-an yang kemudian diikuti dengan perkembangan bahasa pemrograman Secara aktual, penggunaan bahasa pemrograman berorientasi objek pada saat itu masih terbatas, namun telah banyak menarik perhatian di saat itu.[9]

Sekitar lima tahun setelah Smalltalk berkembang, maka berkembang pula metode pengembangan berorientasi objek. Metode yang pertama diperkenalkan oleh Sally Shlaer dan Stephen Mellor (Shlaer-Mellor, 1988) dan Peter Coad dan Edward Yourdon (Coad-Yourdon, 1991), diikuti oleh Grady Booch (Booch, 1991), James R. Rumbaugh, Michael R. Blaha, William Lorensen, Frederick Eddy, William Premerlani (Rumbaugh-Blaha- Premerlani-Eddy-Lorensen, 1991), dan masih banyak lagi. Buku terkenal yang juga berkembang selanjutnya adalah karangan Ivar Jacobson (Jacobson, 1992) yang menerangkan perbedaan pendekatan yang fokus pada *use case* dan proses pengembangan. Sekitar lima tahun kemudian muncul buku yang membahas mengenai metodologi berorientasi objek yang diikuti dengan buku-buku yang lainnya. Di dalamnya juga membahas mengenai konsep, definisi, notasi, terminologi, dan proses mengenai metodologi berorientasi objek.

Karena banyaknya metodologi-metodologi yang berkembang pesat saat itu, maka muncullah ide untuk membuat sebuah bahasa yang dapat dimengerti semua orang. Usaha penyatuan ini banyak mengambil dari metodologimetodologi yang berkembang saat itu. Maka dibuat bahasa yang merupakan gabungan dari beberapa konsep seperti konsep Object Modelling Technique (OMT) dari Rumbaugh dan Booch (1991), konsep The Classes, Responsibilities, Collaborators (CRC) dari Rebecca Wirfs-Brock (1990), konsep pemikiran Ivar Jacobson, dan beberapa konsep lainnya dimana James R. Rumbaigh, Grady Booch, dan Ivar Jacobson bergabung dalam sebuah perusahaan yang bernama Rational Software Corporation menghasilkan bahasa yang disebut dengan Unified Modeling Language (UML). Pada 1996, Object Management Group (OMG)

mengajukan proposal agar adanya standardisasi pemodelan berorientasi objek dan pada bulan September 1997 UML diakomodasi oleh OMG sehingga sampai saat ini UML telah memberikan kontribusinya yang cukup besar di dalam metodologi berorientasi objek dan hal-hal yang terkait di dalamnya.

2.2. 12 Model – View – Controler

Model-View-Controller atau MVC adalah sebuah metode untuk membuat sebuah aplikasi dengan memisahkan data *(Model)* dari tampilan *(View)* dan cara bagaimana 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. Bagian–bagian dari MVC adalah :

1. Model

Model mewakili struktur data. Biasanya model berisi fungsi-fungsi yang membantu seseorang dalam pengelolaan basis data seperti memasukkan data ke basis data, pembaruan data dan lain-lain.

2. View

View adalah bagian yang mengatur tampilan ke pengguna. Bisa dikatakan berupa halaman web.

3. Controller

Controller merupakan bagian yang menjembatani *model* dan *view*. *Controller* berisi perintah-perintah yang berfungsi untuk memproses suatu data dan mengirimkannya ke halaman web.

2.2. 13 Code Igniter

CodeIgniter adalah aplikasi *open source* yang berupa *framework* dengan model MVC (*Model, View, Controller*) untuk membangun *website* dinamis dengan menggunakan PHP. *CodeIgniter* memudahkan *developer* untuk membuat aplikasi web dengan cepat dan mudah dibandingkan dengan membuatnya dari awal. *Framework* secara sederhana dapat diartikan kumpulan dari fungsi-fungsi atau prosedur-prosedur dan *class-class* untuk tujuan tertentu yang siap digunakan

sehingga bisa lebih mempermudah dan mempercepat pekerjaan seorang pemrogram, tanpa harus membuat fungsi atau class dari awal.

Ada beberapa alasan mengapa menggunakan *framework*:

a. Mempercepat dan mempermudah pembangunan sebuah aplikasi web.
b. Relatif memudahkan dalam proses *maintenance* karena sudah ada pola tertentu dalam sebuah *framework* (dengan syarat *programmer* mengikuti pola standar yang ada)
c. Umumnya *framework* menyediakan fasilitas-fasilitas yang umum dipakai sehingga kita tidak perlu membangun dari awal (misalnya validasi, ORM, *pagination, multiple database, scaffolding, pengaturan session, error handling*, dll.
d. Lebih bebas dalam pengembangan jika dibandingkan CMS.

Ada beberapa kelebihan *CodeIgniter* (CI) dibandingkan dengan framework PHP lain yaitu:

a. Performa sangat cepat karena salah satu alasan tidak menggunakan framework adalah karena eksekusinya yang lebih lambat daripada PHP *from the scracth*, tapi *Codeigniter* sangat cepat bahkan mungkin bisa dibilang merupakan framework yang paling cepat dibanding framework yang lain.
b. Konfigurasi yang sangat minim (*nearly zero configuration*) tentu saja untuk menyesuaikan dengan database dan keleluasaan routing tetap diizinkan melakukan konfigurasi dengan mengubah beberapa file konfigurasi seperti *database.php* atau *autoload.php*, namun untuk menggunakan *codeigniter* dengan setting standar, hanya perlu merubah sedikit saja file pada *folder config.*
c. Banyak komunitas dengan banyaknya komunitas CI ini, memudahkan kita untuk berinteraksi dengan yang lain, baik itu bertanya atau teknologi terbaru.
d. Dokumentasi yang sangat lengkap setiap paket instalasi codeigniter sudah disertai user guide yang sangat bagus dan lengkap untuk dijadikan permulaan, bahasanya mudah dipahami.

2.2. 14 Personal Homepage Tool (PHP)

PHP adalah skrip bersifat *server-side* yang ditambahkan ke dalam HTML. Sifat *server-side* berarti pengerjaan skrip dilakukan di *server*, kemudian hasilnya dikirim ke *browser*. Keunggulan dari sifat *server-side* adalah :[10]

1. Tidak diperlukan kompatibilitas *browser* atau harus menggunakan *browser* tertentu, karena serverlah yang mengerjakan skrip PHP.
2. Dapat memanfaatkan sumber-sumber aplikasi yang dimiliki oleh *server*, misalnya koneksi ke *database*.
3. Dapat berkomunikasi dengan layanan-layanan yang menggunakan protokol IMAP, SNMP, NNTP, POP3, HTTP, dan lain-lain.
4. Dapat melakukan koneksi dengan berbagai macam *database*, salah satunya adalah *database* MYSQL Pada awalnya PHP merupakan kependekan dari *Personal Home Page* (Situs personal). PHP pertama kali dibuat oleh Rasmus Lerdorf pada tahun 1995. Pada waktu itu PHP masih bernama *Form Interpreted* (FI), yang wujudnya berupa sekumpulan skrip yang digunakan untuk mengolah data formulir dari *web*. Selanjutnya Rasmus merilis kode sumber tersebut untuk umum dan menamakannya PHP/FI. Dengan perilisan kode sumber ini menjadi sumber terbuka, maka banyak pemrogram yang tertarik untuk ikut mengembangkan PHP. Pada November 1997, dirilis PHP/FI 2.0. Pada rilis ini, *interpreter* PHP sudah diimplementasikan dalam program C. Dalam rilis ini disertakan juga modul modul ekstensi yang meningkatkan kemampuan PHP/FI secara signifikan. Pada tahun 1997, sebuah perusahaan bernama Zend menulis ulang interpreter PHP menjadi lebih bersih, lebih baik, dan lebih cepat. Kemudian pada Juni 1998, perusahaan tersebut merilis *interpreter* baru untuk PHP dan meresmikan rilis tersebut sebagai PHP 3.0 dan singkatan PHP diubah menjadi akronim berulang PHP: *Hypertext Preprocessing.* Pada pertengahan tahun 1999, Zend merilis *interpreter* PHP baru dan rilis tersebut dikenal dengan PHP 4.0. PHP 4.0 adalah versi PHP yang paling banyak dipakai pada awal abad ke-21. Versi ini banyak dipakai disebabkan kemampuannya untuk membangun aplikasi

web kompleks tetapi tetap memiliki kecepatan dan stabilitas yang tinggi. Pada Juni 2004, Zend merilis PHP 5.0. Dalam versi ini, inti dari *interpreter* PHP mengalami perubahan besar. Versi ini juga memasukkan model pemrograman berorientasi objek ke dalam PHP untuk menjawab perkembangan bahasa pemrograman ke arah paradigma berorientasi objek. Penulisan Script PHP Berikut ini adalah contoh penulisan script/kode dari PHP :

```
<HTML>
        <HEAD>
        <TITLE>contoh PHP</TITLE>
        </HEAD>
        <BODY>
        <?php
        echo ”Scripts PHP!”;
        ?>
        </BODY>
</HTML>
```

2.2. 15 HTML (*Hyper Text Markup Language*)

HTML adalah sebuah bahasa *markup* yang digunakan untuk membuat sebuah halaman web dan menampilkan berbagai informasi di dalam sebuahbrowser Internet.*HTML* diciptakan oleh Tim Berners-Lee, seorang penelitiCERN.Berners-Lee mendasarkan HTML pada *Standard Generalized MarkupLanguage*. Dokumen HTML pada dasarnya adalah dokumen teks yangmengandung kode-kode tag yang sesuai dengan spesifikasi HTML. Kode-kodetag itu nantinya diterjemahkan oleh aplikasi browser sehingga dokumen HTMLtadi bisa ditampilkan sesuai dengan yang diinginkan pembuatnya. Secara umum,HTML memiliki empat jenis elemen yaitu:[10]

1. *Structural*, yaitu tanda yang menentukan level atau tingkatan sebuah teks (misalnya sebagai *heading*, paragraf, kutipan, dan sebagainya).

2. *Presentational*, yaitu tanda yang menentukan tampilan sebuah teks (misalnya cetak tebal, miring, garis bawah, dan lain-lain).
3. *Hypertext*, yaitu tanda yang menunjukkan *link* ke bagian lain pada teks tersebut atau ke dokumen lain.

Widget, yaitu tanda yang menghasilkan obyek-obyek tertentu seperti tombol, garis horisontal, dan lain-lain.

2.2. 16 Cascading Style Sheet (CSS)

Cascading Style Sheet (CSS) atau yang biasa disingkat dengan CSS, merupakan suatu dokumen yang digunakan untuk melakuakan pengaturan halaman Web yang ditulis dengan HTML atau XHTML. Penggunaan CSS tidak memerlukan perangkat lunak tertentu karena CSS merupakan *script* yang telah *embedded* dengan HTML. CSS digunakan oleh *Web Designer* untuk menentukan warna, jenis, huruf, tata letak, dan berbagai aspek tampilan dokumen. CSS digunakan terutama untuk memisahkan antara isi dokumen (yang ditulis dengan HTML atau bahasa markup lainnya) dengan presentasi dokumen (yang ditulis dengan CSS). Pemisah ini ditujukan agar dapat mmemisahkan aksesibilitas isi, memberikan lebih banyak keleluasaan dan kontrol terhadap tampilan, dan mengurangi kompleksitas serta pengulangan pada trukur isi.

2.2. 17 Javascript

Javascript merupakan *cross-platform* yang dikembangkan oleh Netscape dan pertama kali digunakan dalam *browser* Netscape. Javascript dibuat agar mudah diintegrasikan kedalam program dan aplikasi lain, misalnya *browser*. Sebagian besar *browser* saat ini sudah mendukung javascript. Oleh karena itu, *script* dari javascript biasanya dimasukkan kedalam suatu HTML dan dieksekusi di*browser* lain.

Javascript bergantung kepada browser(navigator) yang memanggil halaman web yang berisi skrip skrip dari Javascript dan tentu saja terselip di dalam dokumen HTML. Javascript juga tidak memerlukan kompilator atau penterjemah

khusus untuk menjalankannya (pada kenyataannya kompilator Javascript sendiri sudah termasuk di dalam browser tersebut).[7]

2.2. 18 MySQL

MySQL adalah suatu perangkat lunak *database* relasi (*Relational DatabaseManagement system* atau RDBMS), seperti halnya ORACLE, Postgresql, *MSSQL*dan sebagainya.Jangan disalah-artikan dengan SQL.SQL (*Structured QueryLanguage*) sendiri didefinisikan sebagai suatu sintaks perintah-perintah tertentuatau bahasa (program) yang digunakan untuk mengelola suatu *database*.Kepopuleran *MySQL* dimungkinkan karena kemudahannya untukdigunkaan, cepat secara kinerja *query*, dan mencukupi untuk kebutuhan *database*perusahaan-perusahaan skala menengah-kecil.*Database MySQL* merupakan*database* yang menjanjikan sebagai *alternative* pilihan *database* yang dapatdigunakan untuk sistem *database personal* atau organisasi

2.2. 19 Internet

Internet dapat diartikan sebagai jaringan komputer luas dan besar yang mendunia, yaitu menghubungkan pemakai komputer dari suatu negara ke negara lain di seluruh dunia, dimana di dalamnya terdapat berbagai sumber daya informasi dari mulai yang statis hingga yang dinamis dan interaktif. [6]

2.2. 20 Email

Email adalah singkatan dari *Electronic Mail* atau jika dalam bahasa Indonesia adalah surat elektronik. Sebenarnya email merupakan singkatan dari "Electronic mail".Melalui *email* kita dapat mengirim surat elektronik baik berupa teks maupun gabungan dengan gambar, yang dikirimkan dari satu alamat email ke alamat lain di jaringan internet. Seperti layaknya surat biasa pada umumnya, email berfungsi untuk mengirimkan surat atau pesan kepada orang lain. Perbedaan antara surat bisa dengan email adalah email sudah tidak lagi membutuhkan kertas sebagai media untuk menuliskan pesan, media yang di gunakan adalah berupa data digital yang di kirimkan melalui internet.

2.2. 21 Item Based Collaborative Filtering

Item-based collaborative filtering merupakan metode rekomendasi yang didasari atas adanya kesamaan antara pemberian rating terhadap suatu produk dengan produk yang dibeli. Dari tingkat kesamaan produk, kemudian dibagi dengan parameter kebutuhan pelanggan untuk memperoleh nilai kegunaan produk. Produk yang memiliki nilai kegunaan tertinggilah yang kemudian dijadikan rekomendasi. Pada metode ini akan diketahui nilai similaritas antar item dengan tingkat persebaran rating kecil dan nilai similaritas antar item cenderung lebih jarang berubah dibandingkan dengan nilai similaritas antar pengguna. [12]

1. Untuk menghitung similaritas digunakan rumus :

$$\boldsymbol{Sim} = \frac{\mathbf{1}}{(\mathbf{1}+\sum \boldsymbol{Dis})} \qquad \textbf{(2.1)}$$

Keterangan :

Sim : nilai similaritas antar pengguna

$\sum$ Dis : jumlah distance

2. Untuk menghitung distance digunakan rumus :

$$\boldsymbol{Dis} = (\mathbf{N_a} - \mathbf{N_b})^2 \qquad \textbf{(2.2)}$$

Keterangan:

N_a = nilai dari produk orang yang akan diberikan rekomendasi

N_b = nilai dari produk orang yang akan dibandingkan.

Adapun cara penilaian dilakukan dengan memberi nilai 1 untuk pelanggan yang hanya melihat produk dan nilai 2 untuk pelanggan yang membeli produk tersebut. Penghitungan Dis dilakukan berdasarkan jenis produk yang sama.

3. Penghitungan rekomendasi dilakukan dengan rumus:

$$\boldsymbol{R} = \frac{\sum(\boldsymbol{Sim}\times\boldsymbol{N})}{\sum \boldsymbol{Sim}} \qquad \textbf{(2.3)}$$

Keterangan :

R = rekomendasi

∑ = adalah jumlah dari perhitungan Sim x Nilai Produk untuk semua pelanggan.

∑ Sim = jumlah Sim dari pelanggan dengan produk yang memiliki nilai.

2.2. 22 Skala Pengukuran

Skala pengukuran merupakan kesepakatan yang digunakan sebagai acuan untuk menentukan panjang pendeknya interval yang ada dalam alat ukur, sehingga alat ukur tersebut bila digunakan dalam pengukuran akan menghasilkan data kuantitatif. Dengan skala pengukuran ini, maka nilai variabel yang diukur 10 dengan instrumen tertentu dapat dinyatakan dalam bentuk angka, sehingga akan lebih akurat, efisien, dan komunikatif. Berbagai skala yang dapat digunakan untuk penelitian adalah : [11]

1. **Skala *Likert***

Skala *Likert* digunakan untuk mengukur sikap, pendapat, dan persepsi seseorang atau sekelompok orang tentang fenomena sosial.

Dengan skala *Likert*, maka variabel yang akan diukur dijabarkan menjadi indikator variabel. Kemudian indikator tersebut dijadikan sebagai titik tolak untuk menyusun item–item instrumen yang dapat berupa pernyataan atau pertanyaan.

Jawaban setiap item instrumen yang menggunakan skala likert mempunyai gradasi dari sangat positif sampai sangat negatif, yang dapat berupa kata–kata antara lain:

a.Sangat Setuju
b.Setuju
c.Ragu-ragu
d.Tidak Setuju
e.Sangat tidak setuju

a.Selalu
b.Sering
c.Kadang-kadang
d.Tidak pernah

a.Sangat positif
b.Positif
c.Negatif
d.Sangat Negatif

a.Sangat baik
b.Batik
c.Tidak baik
d.Sangat tidak baik

Untuk keperluan analisis kualitatif, maka jawaban itu dapat diberi skor, misalnya :

1. Sangat setuju/selalu/sangat positif diberi skor 5
2. Setuju/sering/positif diberi skor 4
3. Ragu-ragu/kadang-kadang/netral diberiskor 3
4. Tidak setuju/hampir tidak pernah/negatif diberi skor 2
5. Sangat tidak setuju/tidak pernah diberi skor 1

Instrument peneletian yang menggunakan skala likert dapat dibuat dalam bentuk *checklist* ataupun pilihan ganda.

Dalam penyusunan instrumen untuk variabel tertentu, sebaiknya butir-butir pertanyaan dibuat dalam bentuk kalimat positif, netral atau negatif, sehingga responden dapat menjawab dengan serius dan konsisten.

Adapun untuk menghitung prosentase jawaban dapat digunakan rumus :

$$Y = \frac{P}{Q} \times 100\% \qquad \textbf{(2.4)}$$

Dimana

Y : prosentase jawaban responden

P = hasil jumlah skor jawaban responden

Q = jumlah skor ideal untuk seluruh item (jumlah responden x nilai dari skor tertinggi)

Sedangkan untuk menghitung P atau jumlah skor yang diperoleh dari jawaban responden dapat digunakan rumus :

$$P = \sum S_i \times R_i \qquad \textbf{(2.5)}$$

Dimana

P = hasil jumlah skor jawaban responden

S_i = skor untuk jawaban responden ke-i

R_i = banyaknya responden untuk jawaban ke-i