BAB II
TINJAUAN PUSTAKA

2.1 Sistem Informasi

Menurut Gordon B. Davis (1991: 91), sistem informasi adalah suatu sistem yang menerima *input* atau masukan data dan instruksi, mengolah data sesuai dengan instruksi dan mengeluarkan hasilnya.

Menurut Rommey (1997: 16), sistem informasi yang diselenggarakan cara untuk mengumpulkan, memasukkan, mengolah, dan menyimpan data dan terorganisir cara untuk menyimpan, mengelola, mengendalikan dan melaporkan informasi dengan cara yang suatu organisasi dapat mencapai tujuan yang telah ditetapkan.

Menurut Kertahadi (2007), sistem informasi adalah alat untuk menyajikan informasi sedemikian rupa sehingga bermanfaat bagi penerimanya. Tujuannya adalah untuk memberikan informasi dalam perencanaan, memulai, pengorganisasian, operasional sebuah perusahaan yang melayani sinergi organisasi dalam proses mengendalikan pengambilan keputusan.

Sistem Informasi berdasarkan pengertian menurut para ahli dia atas adalah sesuatu system yang menerima data untuk kemudian diolah dan menghasilkan suatu informasi. Tujuan dari penggunaan system informasi adalah untuk memberikan informasi yang dibutuhkan oleh organisasi atau perusahaan yang menggunakannya.

2.2 *Point of Sales*

Pengertian dari *Point of Sale* (POS) menurut Bobby Loardy (2010) yaitu merupakan kegiatan yang berorientasi pada penjualan serta sistem yang membantu proses transaksi. Setiap POS terdiri dari hardware berupa (*Terminal/PC, Receipt Printer, Cash Drawer, Terminal pembayaran, Barcode Scanner*) dan *software* berupa (*Inventory Management*, Pelaporan, *Purchasing, Customer Management*, Standar Keamanan Transaksi, *Return Processing*) dimana kedua komponen tersebut digunakan untuk setiap proses transaksi. *Point of Sales* (POS) dapat berupa sebuah *checkout counters* dalam sebuah toko atau tempat usaha dimana transaksi penjualan terjadi.

2.3 *Hypertext Preprocessor (PHP)*

Menurut Supono dan Putratama (2016:3) mengemukakan bahwa "PHP (PHP: *Hypertext Preprocessor*) adalah suatu bahasa pemrograman yang digunakan untuk menerjemahkan baris kode program menjadi kode mesin yang dapat dimengerti oleh komputer yang berbasis *server-side* yang dapat ditambahkan ke dalam HTML".

Menurut Solichin (2016:11) mengemukakan bahwa "PHP merupakan salah satu bahasa pemrograman berbasis *web* yang ditulis oleh dan untuk pengembang web". PHP merupakan bahasa (*script*) pemrograman yang sering digunakan pada sisi *server* sebuah *web*.

Kumpulan kutipan diatas menerangkan bahwa *hypertext preprocessor* (PHP) merupakan bahasa pemrograman untuk membuat/mengembangkan aplikasi berbasis *web* dan bersifat *open soure* dan ditanamkan ke dalam *script* HTML.

2.4 *Framework CodeIgniter* (CI)

Menurut Budi Raharjo (2015:3), "*CodeIgniter* adalah *framework web* untuk bahasa pemrograman PHP yang dibuat oleh Rick Ellis pada tahun 2006, penemu dan pendiri *EllisLab*. *EllisLab* adalah suatu tim kerja yang berdiri pada tahun 2002 dan bergerak di bidang pembuatan *software* dan *tool* untuk para pengembang web". *CodeIgniter* memiliki banyak fitur (fasilitas) yang membantu para pengembang (*developer*) PHP untuk dapat membuat aplikasi *web* secara mudah dan cepat. Dibandingkan dengan *framework web* PHP lainnya, harus diakui bahwa *CodeIgniter* memiliki desain yang lebih sederhana dan bersifat fleksibel (tidak kaku). *CodeIgniter* mengizinkan para pengembang untuk menggunakan framework secara parsial atau secara keseluruhan

2.5 *Conceptual Data Model (CDM)*

Menurut Ramadhani pada jurnal yang ditulis Oktafiani, *Conceptual Data Model* (CDM) adalah model yang dibuat berdasarkan anggapan bahwa dunia nyata terdiri dari koleksi obyek-obyek dasar yang dinamakan entitas (*entity*) serta hubungan (*relationship*) antara entitas-entitas tersebut. Biasanya CDM direpresentasikan dalam bentuk *Entity Relationship Diagram* (ERD) (Khori, 2018).

Adapun manfaat penggunaan CDM dalam perancangan *database*:

- Memberikan gambaran yang lengkap dari struktur basis data yaitu arti, hubungan, dan batasan-batasan
- Alat komunikasi antar pemakai basis data, *designer*, dan analis.

2.6 *Physical Data Model* (PDM)

Menurut Rahmadani (2006), *Physical Data Model* (PDM) adalah sebuah model dengan menggunakan sejumlah tabel untuk menggambarkan data dan hubungan antar data. PDM adalah model yang menggunakan tabel yang menggambarkan hubungan antara *database* dan hubungan antar tabel serta konsep yang menggambarkan detail dari setiap tabel. PDM merupakan bentuk fisik dari desain *database* yang siap untuk diimplementasikan. PDM menggunakan sekumpulan tabel untuk menggambarkan model yang menggambarkan hubungan antar data. Setiap tabel memiliki beberapa kolom, dan setiap kolom memiliki nama dan tipe data yang unik (Kumbang dan Kusumawati, 2019).

2.7 *MySQL*

Suatu sistem manajemen *database* yang digunakan untuk menyimpan data dalam tabel terpisah dan menempatkan semua data dalam satu gudang besar. Berikut ini akan diuraikan beberapa pengertian *MySQL* menurut para ahli.

Menurut Nugroho (2005) MySQL atau dibaca “My Seequel” adalah sistem basis data relasional atau *Relational Database managemnt System* (RDBMS) yang mana dapat bekerja secara cepat dan mudah digunakan.

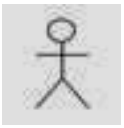
Menurut Sibero (2011:97) *MySQL* atau dibaca dengan “My sekuel adalah suatu RDBMS (*Relational Database Manajemen System*) yaitu aplikasi system yang menjalankan fungsi pengolahan data”.

Berdasarkan penjelasan diatas dapat disimpulkan bahwa, *MySQL* adalah sebuah program *database* yang dapat mengirim ataupun menerima data dan menjalankan fungsi pengolahan data dengan cepat.

2.8 *ICONIX Process*

ICONIX process merupakan suatu metode yang digunakan untuk melakukan pengembangan/pembuatan perangkat lunak. Dalam penggunaannya, *ICONIX process* didukung oleh UML. *Unified Modeling Language* (UML) digunakan sebagai notasi untuk menggambarkan dan mendokumentasikan sistem. Keterkaitan implementasi dengan teknik UML yaitu pengguna dan aksi dapat digambarkan melalui *use case diagram*, objek dalam dunia nyata digambarkan melalui *class diagram*, objek pada *use case* digambarkan melalui *robustness diagram*, interaksi antar objek digambarkan melalui *sequence diagram* serta bagaimana membangunnya digambarkan melalui *class diagram*. (Yulianta & Mursanto, 2008)

Tabel 2.1 Notasi Use Case

No	Gambar	Keterangan
1		Aktor adalah *Abstraction* dari orang atau sistem yang lain yang mengaktifkan fungsi dari target sistem. Untuk mengidentifikasikan harus ditentukan pembagian tenaga kerja dan tugas-tugas yang berkaitan dengan peran pada konteks target sistem.
2		*Use Case* menggambarkan fungsionalitas yang disediakan sistem sebagai unit-unit yang bertukar pesan antar unit dengan aktif yang dinyatakan dengan menggunakan kata kerja.
3		Asosiasi antara aktor dan *use case* yang menggunakan panah terbuka untuk mengindikasikan bila *actor* berinteraksi secara pasif dengan system.
4		*Include*, merupakan di dalam *use case* lain (*required*) atau pemanggilan *use case* oleh *use case* lain, contohnya adalah pemanggilan sebuah fungsi program.
5		*Extend*, merupakan perluasan dari *use case* lain jika kondisi atau syarat terpenuhi.

2.9 *Diagram Use Case*

Diagram *use case* adalah diagram yang menggambarkan hubungan antara aktor dan sistem. *Use case* diagram dapat menggambarkan interaksi antara satu atau lebih aktor dan sistem yang akan dibuat. Diagram *use case* juga dapat digunakan untuk mengetahui fungsi-fungsi yang ada pada sistem, dan juga dapat menunjukkan interaksi aktor dengan sistem. Kemudian komponen ini menggambarkan komunikasi antara aktor dan sistem yang ada.

Oleh karena itu, *use case* dapat disajikan dengan urutan yang sederhana dan mudah dipahami oleh konsumen. Keuntungan dari *use case* itu sendiri adalah dapat meningkatkan komunikasi dengan domain *expert* dan pengguna akhir untuk menentukan pemahaman yang benar tentang persyaratan atau persyaratan sistem.

2.10 *Diagram Robustness*

Robustness diagram adalah gambar objek dari suatu *use case*, yang tujuannya untuk menyempurnakan teks *use case* dan model objek. *Robustness Analysis* digunakan untuk mendapatkan dari use case ke desain mendetail (dan kemudian ke kode), perlu menghubungkan use case ke objek. *Robustness Analysis* membantu menjembatani kesenjangan antara analisis dan desain dengan melakukan hal itu. Ini adalah cara untuk menganalisis teks *use case* dan mengidentifikasi kumpulan objek pertama untuk setiap *use case*. Diagram *robustness* adalah gambar objek dari *use case*.

Diagram *robustness* dan *use case text* harus sesuai dengan tepat, sehingga diagram *robustness* memaksa untuk mengikat *use case text* ke objek. Ini memungkinkan untuk menggerakkan desain berorientasi objek dari *use case*. *Robustness* Diagram digunakan untuk memeriksa kembali bahwa semua kemungkinan tindakan telah dibahas.

2.11 *Diagram Sequence*

Diagram *Sequence* menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan pesan yang dikirimkan dan diterima antar objek. Gambaran *Sequence* Diagram dibuat minimal sebnayak pendefinisan *use*

case yang memiliki proses sendiri atau yang penting semua *use case* yang telah didefinisikan interaksi jalannya pesan sudah dicakup pada *Sequence* Diagram sehingga semakin banyak *use case* yang didefinisikan, maka *Sequence* Diagram yang harus dibuat juga semakin banyak (Rachman, 2018).