**BAB 2**

**TINJAUAN PUSTAKA**

**2.1 Studi Literatur**

Penelitian terkait terdiri dari ringkasan jurnal yang pernah melakukan penelitian dengan topik yang sama. Hasil dari penelitian sebelumnya ditunjukan pada Tabel 2.1

**Tabel 2.1 Penelitian Terkait**

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Judul, Penulis** | **Rangkuman** | **Pembeda** |
| 1 | Pembuatan Aplikasi Presensi Perkuliahan berbasis *fingerprint* Studi kasus : Jurusan Sistem informasi Institut Teknologi Sepuluh Nopember Surabaya, Noval A.M, Febriliyan S dan Radityo P.W | Membahas mengenai sistem yang akan menanggulangi masalah kecurangan presensi yang sering terjadi serta memberikan kemudahan untuk melakukan rekapitulasi presensi mahasiswa selama proses belajar mengajar setiap harinya [5]. | Admin secara berkali (sehari sekali) meng-*update database fingerprint* kedalam *website* |
| 2 | Pencatatan dan Pemantauan Kehadiran Perkuliahan di Lingkungan Politeknik TelkomBerbasis RFID dan Aplikasi Web, Tora Fahrudin | Membahas mengenai model pencatatan kehadiran dengan RFID maupun aplikasi web dan aplikasi pemantauan kehadiran yang berupa *reporting realtime* [6] | Menggunakan surat izin atau surat sakit yang dibawa ke layanan akademik sebagai pengganti ketidakhadiran kuliah |

**2.2 Landasan Teori**

Landasan teori merupakan teori-teori yang ada didalam penulisan naskah

skripsi yang berguna untuk mendukung kesempurnaan penelitian.

**2.2.1 Sistem Informasi**

Sistem adalah suatu sistem di dalam suatu organisasi yang mempertemukan kebutuhan pengolahan transaksi harian, mendukung operasi, bersifat manajerial dan kegiatan strategi dari suatu organisasi dan menyediakan pihak luar tertentu dengan laporan-laporan yang diperlukan [7].

**2.2.2 Pengertian Matrikulasi**

Matrikulasi berarti suatu proses yang membawa sebuah perguruan tinggi dan mahasiswa yang mendaftar untuk kredit perjanjian untuk tujuan mewujudkan tujuan pendidikan siswa. Perjanjian tersebut melibatkan tanggung jawab dari kedua pihak untuk mencapai tujuan tersebut melalui program-program yang didirikan di perguruan tinggi itu meliputi kebijakan, dan persyaratan- persyaratan [8].

**2.2.3 Perancangan Sistem Pemrograman Pendekatan Terstruktur**

Pemrograman dengan metode pendekatan terstuktur adalah konsep atau paradigma atau sudut pandang pemrograman yang membagi-bagi program berdasarkan fungsi-fungsi atau prosedur-prosedur yang dibutuhkan program computer. Modul-modul (pembagian program) biasanya dibuat dengan mengelompokkan fungsi-fungsi dan prosedur-prosedur yang diperlukan sebuah proses tertentu [9].

**2.2.4 *Data Flow Diagram* (DFD)**

*Data Flow Diagram* (DFD) dapat digunakan untuk merepresentasikan sebuah sistem atau perangkat lunak pada beberapa level abstraksi. DFD dapat dibagi menjadi beberapa level yang lebih detail untuk merepresentasikan aliran informasi atau fungsi yang lebih detail. DFD menyediakan mekanisme untuk pemodelan fungsional ataupun pemodelan aliran informasi. Oleh karena itu, DFD lebih sesuai digunakan untuk memodelkan fungsi-fungsi perangkat lunak yang akan diimplementasikan menggunakan pemrograman terstruktur karena pemrograman terstruktur membagi-bagi bagiannya dengan fungs-fungsi dan prosedur-prosedur [9].

DFD tidak sesuai untuk memodelkan sistem perangkat lunak yang akan dibangun menggunakan pemrograman berorientasi objek. Paradigma pemrograman terstruktur dan pemrograman berorientasi objek merupakan hal yang berbeda. Berikut ini notasi-notasi pada DFD menurut Edward Yourdon dan Tom DeMarco dalam Rosa A.S ditunjukkan pada Tabel 2.2 berikut ini :

**Tabel 2.2 Notasi DFD**

|  |  |
| --- | --- |
| **Notasi** | **Keterangan** |
|  | Proses atau fungsi atau prosedur; pada pemodelan perangkat lunak yang akan diimplementasikan dengan pemrograman tersturktur, maka pemodelan notasi inilah yang harusnya menjadi fungsi atau prosedur di dalam kode program.  Catatan :  Nama yang diberikan pada sebuah proses biasanya berupa kata kerja. |
|  | File atau basisdata atau penyimpanan (*storage);* pada pemodelan perangkat lunak yang akan diimplementasikan dengan pemrograman terstruktur, maka pemodelan notasi inilah yang harusnya dibuat menjadi table-tabel basis data *(entity relationship diagram (ERD), Conceptual Data Modem (CDM), Physical Data Model (PDM))*  Catatan : nama yang diberikan pada sebuah penyimpanan biasanya kata benda. |
|  | Entitas luar *(external entity)* atau masukan *(input)*  atau keluaran *(output)* atau orang yang memakai/berinteraksi dengan perangkat lunak yang dimodelkan atau sistem lain yang terkait dengan aliran data dari sistem yang dimodelkan.  Catatan : Nama yang digunakan pada masukan *(input)* atau keluaran *(output)* biasanya berupa kata benda. |
|  | Aliran data merupakan data yang dikirim antar proses, dari penyimpanan ke proses atau dari proses ke masukan *(input)* atau keluaran *(output)*.  Catatan : nama yang digunakan pada aliran data biasanya berupa kata benda. |

Berikut ini adalah tahapan-tahapan perancangn dengan menggunakan DFD :

1. Membuat DFD Level 0 atau sering disebut juga Context Diagram.

DFD level 0 menggambarkan sistem yang akan dibuat sebagai suatu entitas tunggal yang berinteraksi dengan orang maupun sistem lain. DFD level 0 digunakan untuk menggambarkan interaksi antara sistem yang akan dikembangkan dengan entitas luar.

1. Membuat DFD Level 1

DFD Level 1 digunakan untuk menggambarkan modul-modul yang ada dalam sistem yang akan dikembangkan. DFD Level 1 merupakan hasil breakdown DFD Level 0 yang sebelumya sudah dibuat.

1. Membuat DFD Level 2

Modul-modul pada DFD Level 1 dapat di-breakdown menjadi DFD Level 2. Modul mana saja yang harus di-breakdown lebih detail tergantung pada tingkat kedetailan modul tersebut. Apabila modul tersebut sudah cukup detail dan rinci maka modul tersebut sudah tidak perlu untuk di-breakdown lagi. Untuk sebuah sistem, jumlah DFD Level 2 sama dengan jumlah modul pada DFD Level 1 yang di-breakdown.

1. Membuat DFD Level 3 dan seterusnya

DFD Level 3, 4, 5, dan seterusnya merupakan breakdown dari modul DFD Level di-atasnya. Breakdown pada level 3, 4, 5, dan seterusnya aturannya sama persis dengan DFD Level 1 atau level 2.

Pada satu diagram DFD sebaiknya jumlah modul tidak boleh lebih dari 20 buah. Jika lebih dari 20 buah modul, diagram akan terlihat rumit dan susah untuk dibaca sehingga menyebabkan sistem yang dikembangkan juga menjadi rumit.

**2.2.5 ERD (*Entity Relationship Diagram*)**

*Entity Relationship Diagram* merupakan sebuah pemodelan awal basis data yang paling banyak digunakan. ERD dikembangkan berdasarkan teori himpunan dalam bidang matematika. ERD digunakan untuk pemodelan basis data relasional. ERD memiliki beberapa aliran notasi seperti notasi Chen (dikembangkan oleh Peter Chen), Barker (dikembangkan oleh Richard Barker, Ian Palmer, Harry Ellis), notasi Crow’s Foot, dan beberapa notasi lain [9]. Namun yang banyak digunakan adalah notasi dari Chen. Berikut adalah simbol-simbol yang digunakan pada ERD dengan notasi Chen ditunjukkan pada Tabel 2.3 berikut ini :

**Tabel 2.3 Notasi ERD**

|  |  |
| --- | --- |
| **Simbol** | **Deskripsi** |
| Entitas/*Entity*  Nama Entitas | Entitas merupakan data inti yang akan disimpan; bakal table pada basis data; benda yang memiliki data dan harus disimpan datanya agar dapat diakses oleh aplikasi komputer; penamaan entitas biasanya lebih kepada kata benda dan belum merupakan nama table. |
| Atribut | *Field* atau kolom data yang butuh disimpan dalam suatu entitas. |

|  |  |
| --- | --- |
| **Simbol** | **Deskripsi** |
| Atribut kunci primer | *Field* atau kolom data yang butuh disimpan dalam suatu entitas dan digunakan sebagai kata kunci akses *record* yang diinginkan. |
| Atribut multi nilai | *Field* atau kolom data yang butuh disimpan dalam suatu entitas yang dapat memiliki nilai lebih dari satu. |
| Relasi  nama\_relasi | Relasi yang menghubungkan antar entitas; biasanya diawali dengan kata kerja. |
| Asosiasi/ *association* | Penghubung antara relasi dan entitas dimana dikedua ujungnya memiliki *multiplicity* kemungkinan jumlah pemakaian. |

**2.2.6 Metode Pengembangan Sistem**

Metode *waterfall* merupakan metode pengembangan sistem dengan tahap-tahap utama dari model memetakan kegiatan-kegiatan pengembangan dasar. Metode *waterfall* ini sering disebut dengan *classic life cycle* karena tahap demi tahap yang dilalui harus menunggu selesainya tahap sebelumnya dan berjalan beruturan [10]. Metode *waterfall* ditunjukan pada Gambar 2.1 dibawah ini.



**Gambar 2.1. Model *Waterfall***

Metode Waterfalladalah suatu proses pengembangan perangkat lunakberurutan, di mana kemajuan dipandang sebagai terus mengalir ke bawah (sepertiair terjun) melewati fase-fase perencanaan, pemodelan, implementasi (konstruksi),dan pengujian.

Secara garis besar metode waterfallmempunyai langkah-langkah sebagaiberikut [11]:

1) Requirements Analysis and Definition

Tahap ini merupakan tahap analisa terhadap sistem layanan, kendala, dantujuan yang dilakukan melalui konsultasi antara sistem analis denganpengguna sistem terkait kebutuhan tersebut, kemudian ditetapkan secararinci dan berfungsi sebagai spesifikasi sistem.

2) System and Software Design

Tahap ini merupakan proses penuangan pikiran dan perancangan sistemterhadap solusi dari permasalahan yang ada dengan menggunakanperangkat pemodelan sistem seperti diagram alir data (data flow diagram),diagram hubungan entitas (entity relationship diagram) serta struktur datadan bahasan data. Dan persyaratan baik untuk sistem perangkat keras atauperangkat lunak dengan mendirikan sebuah arsitektur sistem secarakeseluruhan.

3) Implementation and Unit Testing

Pada tahap ini, perancangan perangkat lunak direalisasikan sebagaiprogram atau unitprogram, tahap ini dilakukan oleh programmeryangakan menerjemahkan (coding) hasil system and software design. Setelahproses pengkodean selesai maka langkah selanjutnya adalah dilakukannyatestingterhadap sistem yang telah dibuat. Dengan tujuan menemukankesalahan-kesalahan yang terdapat pada sistem tersebut.

4) Integration and System Testing

*Unit* program individu atau program diintegrasikan dan diuji sebagaisistem yang lengkap untuk memastikan bahwa persyaratan perangkatlunak telah dipenuhi. Setelah pengujian, sistem *software* diserahkankepada pelanggan.

5) Operation and Maintenance

Ini adalah fase yang terpanjang. Sistem ini dipasang dan dimasukan kedalam penggunaan praktis. pemeliharaan melibatkan mengoreksikesalahan yang tidak ditemukan dalam tahap awal siklus, meningkatkanimplementasi unitsistem dan peningkatan sistem sebagai kebutuhan baruditemukan.