BAB II
TINJAUAN PUSTAKA DAN LANDASAN TEORI

2.1. Tinjauan Pustaka

Penelitian mengenai sistem informasi monitoring imunisasi pernah dilakukan oleh LM. Fajar Israwan, dkk (2022). Permasalahan pada penelitian ini yaitu pencatatan imunisasi direkap pada Kartu Menuju Sehat (KMS) yang rentang hilang dan sobek sehingga orang tua tidak mempunyai riwayat data imunisasi anaknya. tujuan penelitian ini untuk merancang bangun sistem informasi monitoring data imunisasi berkala pada posyandu berbasis android [2].

Penelitian serupa dilakukan oleh Lela Triana, dkk (2020). Pada sistem berjalan saat ini data imunisasi anak dicatat di Katu Menuju Sehat (KMS) yang rentan hilang dan orang tua tidak mempunyai riwayat imunisasi anakanya. Aplikasi monitoring imunisasi data imunisasi merupakan aplikasi yang dibangun untuk membantu mengorganisasikan data menjadi informasi yang diperlukan dalam mendukung kegiatan monitoring secara berkala pada Puskesmas Pembina [3].

Penelitian sebelumnya dilakukan oleh Reza Rizki Fauzi, (2022). Kegiatan posyandu saat ini masih menggunaka metode catatan terutama pada data-data kegiatan posyandu. Untuk menunjang kelancaran aktivitas maka dibuatlah sistem informasi posyandu Parakansalak Sukabumi menggunakan metode *prototype* [4].

Penelitian sebelumnya dilakukan oleh Ahmad Chusyairi, dkk (2019). Salah satu prioritas wajib dalam pembangunan di Kabupaten Banyuwangi adalah Kesehatan, khususnya *Continuum of Care Life Cycle* untuk dapat mewujudkan genarasi Banyuwangi Emas, yaitu : Kesehatan bayi dan balita. Tujuan penelitian ini adalah percepatan Upaya Kesehatan Bersumberdaya Masyarakat (UKBM) [5].

Penelitian sebelumnya dilakukan oleh Kenny Aptha Pratama, dkk (2020). Posyandu Anggrek ini selaku pelayanan masyarakat masih mempunyai banyak kelemahan dalam proses pendaftaran, keakuratan data juga kurang dapat diterima. Dengan dukungan teknologi informasi yang ada, pengolahan data yang dilakukan saat ini dapat digantikan dengan sistem informasi berbasis website [6].

Berdasarkan perbedaan dari kajian penelitian diatas, maka peneliti akan membuat Sistem Informasi Monitoring Imunisasi Balita Di Puskesmas Cilacap Tengah 2. Sitem informasi ini dibuat menggunakan

metode *waterfall*, native, bahasa pemrogaman PHP, Database Developer MySql, Web Server Apache dan tools editor Visual Studio Code. Terdapat fitur wmail gateaway, untuk memberikan notifakasi serta pengingat jadwal imunisasi untuk bayi yang akan dilaksanakan. Hasil dari penelitian ini adalah sebuah sistem yang dapat memudahkan admin dan bidan dalam menginputkan data-data imunisasi, serta memudahkan orang tua dalam memonitoring imunitasi untuk anaknya. Selain itu, sistem ini memudahkan Kepala Puskesmas dalam melihat rekapan imunisasi di Puskesmas Cilacap Tengah 2.

Tabel 2. 1 Perbedaan Penelitian Sebelumnya

No	Judul	Peneliti	Persamaan	Perbedaan
1.	Penerapan Sistem Informasi pada Monitoring Imunisasi Berkala Posyandu Berbasis Android.	LM. Fajar Israwan, Jabal Nur, Yani	Metode pengembangan yang digunakan yaitu metode *waterfall*.	**Penelitian sebelumnya,** berbasis android **Penelitian yang dirancang,** berbasis website
2.	Aplikasi Monitoring Data Imunisasi Berkala Untuk Meningkatkan Pelayanan Posyandu Menggunakan Metode RAD Berbasis Android	Lela Triana, Ria Andryani, Kurniawan	Pengujian sistem menggunakan *Blackbox testing*.	**Penelitian sebelumnya** menggunakan metode RAD berbasis android. **Penelitian yang sedang dirancang** menggunakan metode *waterfall* berbasis website

3.	Perancangan Sistem Informasi Monitoring Kegiatan Posyandu Parakansalak Sukabumi	Reza Rizki Fauzi	Berbasis *website*.	**Penelitian sebelumnya** menggunakan metode *prototype*. **Penelitian yang sedang dirancang** menggunakan metode *waterfall.*
4.	Rancang Bangun Sistem Informasi Kesehatan Bayi dan Balita Berbasis Android.	Ahmad Chusyairi, Pelsri Ramadar Noor Saputra	Pengujian sistem menggunakan *Blackbox testing*.	**Penelitian sebelumnya** menggunakan metode *agile,* berbasis *android*. **Penelitian yang sedang dirancang** menggunakan metode *waterfall,* berbasis *website*.
5.	Sistem Informasi Pendaftaran Imunisasi Posyandu Anggrek di Desa Tanjung Medan Berbasis Web.	Kenny Aptha Pratama, Marnis Nasution, Musthafa Haris Munandar	Pengujian sistem menggunakan *Blackbox testing*, berbasis website, pengembangan sistem menggunakan metode *waterfall*.	**Penelitian sebelumnya** tidak menggunakan *email gateaway*. **Penelitian yang sedang dirancang** menggunakan *email gateaway*.

2.2. Landasan Teori

2.2.1. Sistem Informasi

Sistem adalah suatu jaringan kerja dari prosedur-prosedur yang saling berhubungan, berkumpul bersama-sama untuk melakukan suatu kegiatan tertentu untuk mencapai tujuan tertentu. Sedangkan informasi adalah sekumpulan data-data yang awalnya tidak memiliki sebuah nilai atau tidak bermanfaat bagi orang lain, namun setelah diolah menjadi sebuah bentuk kesatuan yang baru dan bentuk tersebut memiliki nilai yang dapat digunakan dalam suatu pengambilan keputusan.

Sistem Informasi atau disingkat SI merupakan serangkaian unsur-unsur atau komponen-komponen yang saling berhubungan dan memiliki tugas yaitu mengumpulkan, menyimpan, memproses dan mendistribusikan suatu informasi yang nantinya dapat digunakan sebagai bahan landasan bagi pengambilan keputusan [7].

2.2.2. Rekayasa Perangkat Lunak

Pengertian Rekayasa Perangkat Lunak adalah suatu disiplin ilmu yang membahasa tentang semua aspek produksi perangkat lunak, mulai dari tahap awal yaitu analisi kebutuhan pengguna, menentukan spesifikasi dari keutuhan pengguna, desain, pengkodean, pengujian, sampai pemeliharaan sistem setelah dikembangkan. Perangkat lunak adalah seluruh perintah yang digunakan untuk memproses informasi. Perangkat lunak dapat berupa program atau prosedur. Program merupakan kumpulan perintah yang dimengerti oleh komputer sedangkan prosedur merupakan perintah yang dibutuhkan oleh pengguna dalam memproses informasi [8].

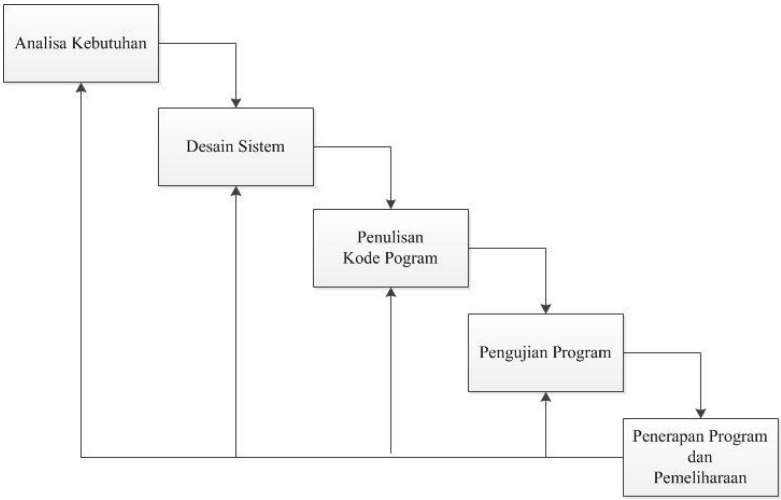
2.2.3. Metode *Waterfall*

Metode *waterfall* merupakan metode yang sering digunakan oleh analisis sistem dalam pengerjaan dari suatu sistem yang dilakukan secara berurutan/linier (Kadir : 2003) [9]. Metode pengembangan *waterfall* mempunyai keunggulan dalam membangun dan mengembangkan suatu sistem antara lain:

1) Kualitas dari sistem yang dihasilkan akan baik, karena pelaksanaanyya dilakukan secara bertahap sehingga tidak terfokus pada atahapan tertentu.
2) Dokumen oengembangan sistem sangat terorganisir, karena setiap fase harus terselesaikan dengan lengkap sebelum melangkah ke fase

berikutnya. Jadi setiap fase atau tahapan akan mempunyai dokumen tertentu.

Langkah-langkah pengembangan sistem *waterfall* dapat dilihat pada **Gambar 2.1.**



Gambar 2. 1 Metode Waterfall Menurut Abdul Kadir

Penjelasan dari **Gambar 2.1** sebagai berikut:

1) Analisa Kebutuhan
   Langkah ini merupakan Analisa terhadap kebutuhan sistem. Pengumpulan data dalam tahap ini bisa dilakukan dengan sebuah penelitian, wawancara atau studi literatur. Tahapan ini akan menghasilkan dokumen *user requirement* atau bisa dikatakan sebagai data yang berhubungan dengan keinginan *user* dalam pembuatan sistem.
2) Desain Sistem
   Tahapan dimana dilakukan penuangan pikiran dan perancangan sistem terhadap solusi dari permaalahan yang ada dengan menggunakan perangkan pemodelan sistem seperti alir data (*data*

flow diagram), diagram hubungan entitas (*entity relationship diagram*) serta struktur dan bahasan data.

3) Penulisan Kode Program
   Penulisan kode program atau coding merupakan penerjemahan design dalam bahasa yang bisa dikenali oleh komputer. Tahapan inilah yang merupakan tahapan secara nyata dalam mengerjakan suatu sistem. Dalam artian penggunaan komputer akan dimaksimalkan dalam tahap ini. Setelah pengkodean selesai maka akan dilakukan testing terhadap sistem yang telah dibuat. Tujuan testing adalah menemukan kesalahan-kesalahan terhadap sistem tersebut dan kemudian bisa diperbaiki.
4) Pengujian Program
   Tahap akhir dimana sistem yang baru diuji kemampuan dan keefektidannya sehingga didapatkan kekurangan dan kelemahan sistem yang kemudian dilakukan pengkajian ulang dan perbaikan terhadap aplikasi menjadi lebih baik dan sempurna.
5) Penerapan Program dan Pemeliharaan
   Penerapan program dan pemeliharaan dapat dilakukan secara beriringan saat diimplementasikan.

2.2.4. Black Box Texting

Metode *BlackBox Testing* adalah sebuah metode yang dipakai untuk menguji sebuah software tanpa harus memperhatikan detail software. Proses *BlackBox Testing* dengan cara mencoba program yang telah dibuat dengan mencoba memasukka data pada tiap formnya. Pengujian ini diperlukan untuk mengetahui program tersebut berjalan sesuai dengan yang dibutuhkan.

Metode *BlackBox Testing* merupakan salah satu metode yang mudah digunakan karena hanya memerlukan batas bawah dan batas atas dari data yang diharapkan. Estimasi banyaknya data uji dapat dihitung melalui banyaknya field data entri yang akan diuji, aturan entri yang harus dipenuhi serta kasus batas atas dan batas bawah yang memenuhi. Dengan metode ini dapat diketahui jika fungsionalitas masih dapat menerima masukan data yang tidak diharapkan makan menyebabkan data yang disimpan kurang valid [10].

Pengujian *black-box* memiliki dua jenis pengujian yaitu pengujian fungsional dan pengujian *non* fungsional. Pengujian *black-box* (fungsionalitas) menguji bug hanya berdasarkan kegagalan fungsi perangkat lunak yang terungkap dalam bentuk output yang salah. *Black Box* Testing cenderung untuk menemukan hal-hal berikut yaitu:

1) Fungsi yang tidak benar atau tidak ada.
2) Kesalahan antarmuka (*interface errors*).
3) Kesalahan pada struktur data dan akses basis data.
4) Kesalahan performansi
5) Kesalahan inisialisasi dan terminasi.

Keuntungan utama pengujian *Black Box* adalah :

1) Sumber daya yang dibutuhkan yang relatif lebih sedikit dibandingkan dengan pengujian *white box*
2) Efektivitas sumber daya dapat dilakukan dengan pengujian secara otomatis maka berkontribusi pada periode pengujian yang lebih singkat.
3) Kemampuan untuk melakukan hampir semua kelompok *test case*, seperti *availability* (*response time*) *reability*, *load durability* dan kelompok pengujian yang terkait dengan *operation*, *revision*, dan *transition factors*.

2.2.5. Pemrogaman Berorientasi Objek (PBO)

Pemrograman Berorientasi Objek (PBO) atau *Object Oriented Programming* (OOP) merupakan suatu pendekatan pemrograman yang menggunakan *object* dan *class*. OOP memberikan kemudahan dalam pembuatan sebuah program, keuntungan yang didapat apabila membuat PBO atau OOP antara lain:

1) *Reusability*, kode yang dibuat dapat digunakan kembali.
2) *Extensibility*, pemrogram dapat membuat metode baru atau mengubah yang sudah ada sesuai yang diinginkan tanpa harus membuat kode dari awal.
3) *Maintainability*, kode yang sudah dibuat lebih mudah untuk dikelola apabila aplikasi yang dibuat berskala besar yang memungkinkan adanya error dalam pengembangannya hal tersebut dapat diatasi dengan OOP karena pemrograman OOP yang lain yaitu : Alamiah, dapat diandalkan (*reliable*), dapat dipakai kembali (*reusable*), mudah dirawat (*maintainable*), dapat diperluas (*extendable*),efisiensi waktu.

Objek dalam OOP adalah unit terkecil pemrograman yang masih memiliki data (sifat karakteristik) dan fungsi. Objek merupakan entitas dari sebuah keadaan, peilaku dan identitas yang tugasnya dirumuskan dalam suatu lingkup masalah, pendeklarasian objek dari sebuah class disebut dengan instance. *Class* adalah wadah yang berisi pemodelan suatu objek, mendeskripsikan karakteristik dan fungsi objek tersebut. Karena *class* merupakan wadah yang digunakan untuk menciptakan suatu objek, maka *class* harus dibuat terlebih dahulu.

2.2.6. Struktur Data

Struktur data adalah cara menyimpan atau merepresentasikan data di dalam computer agar bisa dipakai secara efisien. Berikut adalah bagian-bagian yag ada pada struktur data:

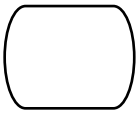
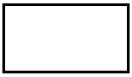
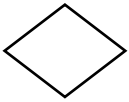
A. Flowchart

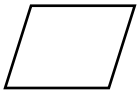
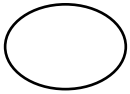
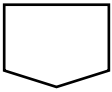
Flowchart adalah sebuah diagram yang menjelaskan alur dari proses sebuah program. Dalam pembuatan sebuah program, *flowchart* berguna untuk menerjemahkan proses berjalannya sebuah program agar mudah dipahami. Setiap langkah digambarkan dengan garis atau arah panah. *Flowchart* terdiri dari 5 jenis, yang masing-masing mempunyai karakteristik dalam penggunaanya, diantara lain yaitu :

1) *Flowchart* dokumen
2) *Flowchart* program
3) *Flowchart* proses
4) *Flowchart* sistem
5) *Flowchart* skematik

Pada dasarnya simbol-simbol dalam *flowchart* mempunyai arti masing-masing. Berikut adalah simbol-simbol yang sering digunakan dalam proses pembuatan *flowchart* seperti yang terdapat pada **Tabel 2.2** Simbol Flowchart.

Tabel 2. 2 Simbol Flowchart

No	Simbol	Nama Simbol	Fungsi
1.		*Terminator*	Simbol untuk menunjukan awal atau akhir dari aliran proses. Biasanya menggunakan kata-kata '*Start', 'End'*, 'Mulai', 'Selesai'.
2.		*Garis Alir (Flow Line)*	Tanda panah yang menunjukan arah aliran dari suatu proses ke proses yang lain
3.		*Process*	Simbol untuk menunjukan sebuah langkah proses atau operasi. Umumnya, menggunakan kata kerja dalam deskripsi yang singkat dan jelas.
4.		*Decision*	Simbol untuk menunjukan sebuah langkah pengambilan keputusan. Umumnya, menggunakan bentuk pertanyaan dan biasanya jawabannya terdiri dari '*Yes'* dan '*No'* atau 'Ya' dan 'Tidak' yang menentukan alur dalam *flowchart* berjalan selanjutnya berdasarkan kriteria atau pertanyaan tersebut.
5.		*Document*	Simbol untuk menunjukan proses atau keberadaan dokumen.

6.		*Input/Output*	Simbol untuk menunjukan data yang menjadi *input* atau *ouput* proses.
7.		*Connector (On-page)*	Simbol untuk menunjukan hubungan simbol dalam *flowchart* sebagai pengganti garis untuk menyederhanakan bentuk saat simbol yang akan dihubungkan jaraknya berjauhan dan rumit jika dihubungkan dengan garis.
8.		*Off-page Connector*	Fungsinya sama dengan *connector*, akan tetapi digunakan untuk menghubungkan simbol simbol yang berada pada halaman yang berbeda. Label untuk *connector* dapat menggunakan huruf dan *Off Page Connector* menggunakan angka.

B. Unified Modeling Languange (UML)

Unified Modeling Languange merupakan salah satu metode pemodelan visual yang digunakan dalam perancangan dan pembuatan sebuah software yang berorientasi pada objek. UML merupakan sebuah standar penulisan atau semacam blue print dimana didalamnya termasuk sebuah bisnis proses, penulisan kelas-kelas dalam sebuah bahasa yang spesifik. Terdapat beberapa diagram UML yang diguanakn dalam pengembangan sebuah sistem yaitu:

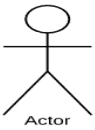
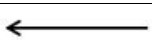
a. Use Case

Use case adalah sebuah interaksi yang saling berkaitan antara actor dengan sistem. *Use case* diagram adalah proses penggambaran yang

dilakukan untuk menunjukkan hubungan antara pengguna dengan sistem yang sedang dirancang.

Terdapat 2 elemen penting dalam *use case* diagram, yaitu Actor dan UC. Actor adalah segala sesuatu yang berinteraksi langsung dengan sistem. Actor dinotasikan dengan simbol gambar orang-orangan *(stick-man)* dengan nama kata benda di bagian bawah yang menyatakan peran atau sistem. *Use case* dinotasikan dengan simbol elips dengan nama kata kerja aktif di bagian dalam yang menyatakan aktivitas dari perspektif actor, seperti pada **Tabel 2.3** Simbol Use Case di bawah ini.

Tabel 2. 3 Simbol Use Case

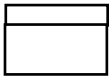
No	Gambar	Nama Gambar	Keterangan
1.		*Actor*	Menspesifikasikan himpunan peran yang pengguna mainkan ketika berinteraksi dengan *use case*
2.		*Dependency*	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri (*independent*) akan mempengaruhi elemen yang bergantung pada elemen yang tidak mandiri (*dependent*).
3.		*Generalization*	Hubungan dimana objek anak (*descendent*) berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk
4.		*Include*	Menspesifikasikan bahwa *use case* sumber secara eksplisit

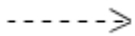
5.		*Extend*	Menspesifikasikan bahwa *use case* target memperluas perilaku dari *use case* sumber pada suatu titik yang diberikan.
6.		*Association*	Apa yang menghubungkan antara objek satu dengan objek lainnya.
7.		*System*	Menspesifikasikan paket yang menampilkan sistem secara terbatas.
8.		*Use Case*	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil terukur bagi suatu aktor.
9.		*Collaboration*	Interaksi aturan-aturan dan elemen lain yang bekerja sama untuk menyediakan perilaku yang lebih besar dari jumlah dan elemen-elemennya (sinergi).
10.		*Note*	Elemen fisik yang eksis saat aplikasi dijalankan dan mencerminkan suatu sumber daya komputasi.

b. Class Diagram

Class diagram adalah sebuah spesifikasi yang jika diinstansiasi akan menghasilkan sebuah objek dan merupakan inti dari pengembangan dan desain berorientasi objek. *Class* menggambarkan keadaan (atribut/properti) suatu sistem, sekaligus menawarkan layanan untuk memanipulasi keadaan tersebut (metoda/fungsi). Seperti yang dapat dilihat pada **Tabel 2.4** Simbol Class Diagram di bawah ini.

Tabel 2. 4 Simbol Class Diagram

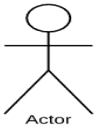
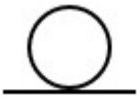
No	Gambar	Nama Gambar	Keterangan
1.		*Generalization*	Hubungan dimana objek anak (*descendent*) berbagi perilaku dan struktur data dari objek yang ada diatasnya objek induk (*ancestor*)
2.		*Nary Association*	Upaya untuk menghindari asosiasi dengan lebih dari 2 objek
3.		*Class*	Himpunan dari objek-objek yang berbagi atribut serta operasi yang sama
4.		*Collaboration*	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu actor
5.		*Realization*	Operasi yang benar-benar dilakukan oleh suatu objek

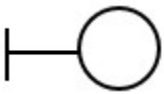
6.		*Dependency*	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri (*independent*) akan mempengaruhi elemen yang bergantung pada elemen yang tidak mandiri
7.		*Association*	Apa yang menghubungkan antara objek satu dengan objek lainnya

c. Sequence Diagram

Sequence Diagram adalah tool yang sering digunakan dalam pengembangan sistem informasi secara *object-oriented* untuk menampilkan interaksi antar objek. *Sequence Diagram* digunakan sebagai perkakas dalam perancangan antarmuka pemakai. *Sequence Diagram* lebih ditujukan untuk memperlihatkan semua bagian/divisi pada sebuah organisasi yang terlibat dalam proses bisnis. Berikut beberapa simbol Sequence Diagram terdapat pada **Tabel 2.5** Simbol Sequence Diagram.

Tabel 2. 5 Simbol Sequence Diagram

No	Gambar	Nama Gambar	Keterangan
1.		*Actor*	Menggambar orang yang sedang berinteraksi dengan sistem.
2.		*Entity Class*	Menggambarkan hubungan yang akan dilakukan.

3.		*Boundary Class*	Menggambarkan sebuah gambaran dari *form*
4.		*Control Class*	Menggambarkan penghubung antara *boundary* dengan tabel
5.		*A Focus of Control & A Life Line*	Menggambarkan tempat mulai dan berakhirnya *massage*
6.		*A Massage*	Menggambarkan Pengiriman Pesan

2.2.7. Basis Data

Basis data adalah kumpulan data yang saling berhubungan secara logis dan didesain untuk mendapatkan data yang dibutuhkan oleh suatu organisasi. Basis data merupakan komponen penting dalam sistem informasi modern. *Database* merupakan suatu kumpulan data terhubung (*integrated*) yang disimpan dengan cara tertentu sehingga mudah untuk digunakan sehingga proses modifikasi data dapat dilakukan dengan mudah dan terkontrol. *Database system* mempunyai elemen penting yaitu:

1) *Database* sebagai inti dari sistem basis data
2) Program aplikasi untuk manajemen basis data
3) Perangkat keras sebagai penunjang operasi manajemen data
4) *Brainware* yang mempunyai peran penting dalam sistem tersebut

MySQL dikembangkan oleh suatu perusahaan yang terletak di Swedia yang bernama MySQL AB, yang pada saat itu Bernama TcX Data Konsult AB sekitar tahun 1994-1995. My SQL sudah ada sejak 1979 termasuk jenis *Relation Database Management System* (RDBMS) digunakan oleh banyak batas internet sebagai basis data dari informasi yang ditampilkan pada situs web. Terkenalnya MySQL dimungkinkan karena mudahnya untuk digunakan cepat secara kinerja *query*, dan

mencukupi kebutuhan basis data terhadap perusahaan skala menengah dan kecil. Sebuah basis data terdapat pada MySQL yang mengandung satu atau beberapa tabel yang berisi sejumlah baris dan kolom.

MySQL merupakan salah satu turunan konsep utama dalam database, yaitu SQL (*Structured Query Language*). SQL adalah bahasa yang digunakan dalam mengakses data di sebuah database relasional. SQL sering dikenal dengan istilah query, dan bahasa SQL secara prakteknya digunakan sebagai bahasa standar dalam manajemen database relasional. Dalam penggunaan SQL terdapat beberapa perintah yang bertujuan untuk mengakses dan memanajemen data yang terdapat dalam database. Secara garis besar, SQL Server terdapat 3 jenis perintah yaitu:

1. Data Definition Language (DDL)

DDL merupakan sub perintah dari bahasa SQL yang digunakan untuk membangun rangkaian sebuah database. Terdapat tiga perintah penting dalam DDL, yaitu :

a) *CREATE*, merupakan perintah yang digunakan untuk membuat, seperti membuat database baru, tabel baru, dan kolom baru. Contoh : *CREATE TABEL* nama_tabel.
b) *ALTER*, merupakan perintah yang digunakan untuk mengubah struktur tabel yang telah dibuat. Didalamnya terdapat menambah kolom, menghapus kolom, mengubah nama tabel, dan memberikan atribut pada kolom. Contoh : *ALTER TABEL* nama_tabel *ADD* nama_kolom *datatype*.

2. Data Manipulation Language (DML)

DML adalah sub perintah dari bahasa SQL yang digunakan untuk memanipulasi data dalam database yang telah dibuat. Terdapat 4 (Empat) perintah penting dalam DML, yaitu :

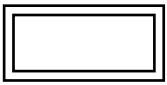
a) *INSERT*: perintah ini digunakan untuk memasukkan data baru ke dalam sebuah tabel. Perintah ini tentu saja bisa dijalankan ketika database dan tabel sudah dibuat. Contoh: *INSERT INTO* nama_tabel *VALUES* (data1, data2, dst…);
b) *SELECT*: perintah ini digunakan untuk mengambil dan menampilkan data dari tabel atau bahkan dari beberapa tabel dengan penggunaan relasi. Contoh: *SELECT* nama_kolom1, nama_kolom2 *FROM* nama_tabel;
c) *UPDATE*: perintah update digunakan untuk memperbaharui data pada sebuah tabel. Contoh: *UPDATE* nama_tabel *SET* kolom1=data1, kolom2=data2,… *WHERE* kolom=data;

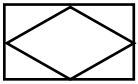
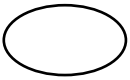
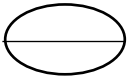
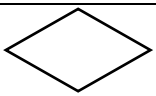
DELETE: perintah delete digunakan untuk menghapus data dari sebuah tabel. Contoh: *DELETE FROM* nama_tabel *WHERE* kolom=data.

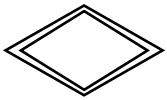
3. Entity Relationship Diagram (ERD)

ERD adalah suatu rancangan atau bentuk hubungan suatu kegiatan di dalam sistem yang berkaitan langsung dan mempunyai fungsi di dalam proses tersebut. ERD adalah suatu pemodelan dari basis data relasional yang didasarkan atas persepsi di dalam dunia nyata, dunia ini senantiasa terdiri dari sekumpulan objek yang saling berhubungan antara satu dengan yang lainnya. Suatu objek disebut *entity* dan hubpeungan yang dimilikinya disebut *relationship*. Suatu *entity* bersifat unik dan memiliki atribut sebagai pembeda dengan entity lainnya.

Tabel 2. 6 Simbol ERD

No	Gambar	Nama Gambar	Keterangan
1.		*Entity*	Suatu *entity* digambarkan sebagai sebuah persegi panjang yang memiliki nama entity tersebut.
2.		*Weak Entity*	Suatu *entity* yang tidak dapat diidentifikasikan melalui atributnya dengan sendirinya. Keberadaan *weak entity* bergantung kepada entity lain yang disebut *owner entity*.

3.		*Associative Entity*	Entity yang digunakan pada *many to many relationship*.
4.		*Attribute*	Dalam notasi *Chen*, sebuah atribut digambarkan sebagai sebuah oval yang memuat nama atribut tersebut.
5.		*Key Attribute*	Suatu atribut yang mengidentifikasi suatu *entity* dengan sangat spesifik atau unik. Nama dalam *Key Attribute* selalu di *underscore*
6.		*Multivalued Attribute*	*Attribute* yang dapat memuat lebih dari satu nilai *(Multivalued)*. *Multivalued Attribute* digambarkan dengan dua oval.
7.		*Strong Relationship*	Hubungan dimana sebuah keberadaan entity bergantung dengan entity lain, dan *Primary Key* dari *Child Entity* tidak

			memuat komponen PK *Parent Entity*.
8.		*Weak Relationship*	Suatu relationship dimana keberadaan *Child entity* bergantung pada induknya, dan PK *Child entity* memuat komponen PK *Parent entity*.

Komponen dalam menyusun ERD antara lain sebagai berikut :

a) Entitas merupakan suatu objek dalam dunia nyata yang bisa dibedakan dengan objek lain, sebagai contoh pengelola, dokter dan pemilik. Entitas tersebut terdiri dari beberapa atribut sebagai contoh : nama, alamat, umur dan lain sebagainya.

b) Atribut merupakan entitas pasti yang memiliki elemen yang berfungsi untuk dapat mendeskripsikan karakteristik dari suatu entitas tersebut seperti contoh di atas. Isi dari atribut tersebut memiliki sesuatu yang biasa mengidentifikasikan isi elemen yang satu dengan yang lainnya. Terdapat dua jenis Atribut antara lain sebagai berikut :

1. *Identifier* (*key*) yang berfungsi sebagai penentu *entity* secara unik (*primary key*).
2. *Descriptor* (*nonkey attribute*) digunakan untuk dapat menspesifikasikan karakteristik dari sebuah *entity* yang tidak unik.

c) Relasi adalah suatu hubungan antara beberapa entitas himpunan relasi antar entitas pemetaan kardinalitas terdiri dari :

1. One-to-one

Setiap baris data pada tabel pertama dihubungkan hanya ke satu baris data pada tabel ke dua. Hubungan antara *file* pertama dan *file* kedua adalah satu berbanding satu.

2. One-to-many

Setiap baris data dari tabel pertama dapat dihubungkan ke satu baris atau lebih data pada tabel ke dua. Hubungan antara *file* pertama dan *file* kedua adalah satu berbanding banyak atau banyak berbanding satu.

3. Many-to-one

Kebalikan dari *relation One To Many* dimana setiap baris data dari tabel pertama dihubungkan lebih dari satu baris ke tabel kedua. Hubungan antara *file* pertama dan *file* kedua adalah banyak berbanding.

2.2.8. Monitoring

Monitoring (bahasa Indonesia : pemantauan) adalah pemantauan yang dapat dijelaskan sebagai kesdaran tentang apa yang ingin diketaui, pemantauan berkadar tingkat tinggi dilakukan agar dapat membuat pengukuran melalui waktu yang menunjukkan pergerakan kea rah tujuan atau menjauh dari itu. Monitoring akan memberikan informasi tentang status dan kecenderungan bahwa pengukuran dan evaluasi yang diselesaikan berulang dari waktu k eke waktu, pemantauan umumnya dilakukan untuk tujuan tertentu, untuk memeriksa terhadap proses berikut objek atau untuk mengevaluasi kondisi atau kemajuan menuju tujuan hasil manajemen atas efek Tindakan dari beberapa jenis antara lain tindakan untuk mempertahankan manajemen yang sedang berjalan.

Kegiatan monitoring bisa diartikan sebagai suatu kegiatan memonitor atau mengawasi seluruh aktivitas yang dilakukan oleh seseorang. Kegiatan monitoring dapat dilakukan secara langsung dengan cara peninjauan langsung terhadap aktivitas yang sedang berlangsung seperti peninjaun barang yang masuk, barang yang keluar dan lain-lain. Sedangkan monitoring tidak langsung dilakukan melalui kegiatan penelaahan laporan tertulis, mencermati lapotan lisan atau wawancarai salah satu dari beberapa orang yang terlibat dalam satu kegiatan [11].

2.2.9. Imunisasi

Imunisasi merupakan usaha memberikan kekebalan pada balita dengan memasukkan vaksin ke dalam tubuh agar tubuh membuat zat antibodi untuk mencegah terhadap penyakit tertentu. Proses pembentukan antibodi untuk melawan antigen secara alamiah disebut imunisasi alamiah, sedangkan program imunisasi melalui pemberian vaksin adalah upaya stimulasi terhadap sistem kekebalan tubuh untuk menghasilkan antibodi dalam upaya melawan penyakit dengan melumpuhkan antigen

yang telah dilemahkan yang berasal dari vaksi. Sedangkan yang dimaksud vaksin adalah bahan yang dipakai untuk merangsang pembentukan zat antibodi yang dimasukkan ke dalam tubuh melalui suntikan. Tujuan pemberian imunisasi adalah balita menjadi lebih kebal terhadap penyakit yang dapat dicegah dengan imunisasi sehingga dapat menurunkan angka morbiditas dan mortalitas serta mengurangi kecacatan akibat penyakit tertentu [12].

Pemberian imunisasi disesuaikan dengan usia anak dengan jarak waktu antara imunisasi yang satu dengan yang lainnya yaitu 4 minggu atau 1 bulan. Untuk imunisasi dasar lengkap, bayi berusia 0 – 1 bulan diberikan imunisasi Hepatitis B (HB-0), bayi berusia 1 bulan diberikan imunisasi BCG dan Polio 1, bayi berusia 2 bulan diberikan imunisasi DPT 1 dan Polio 2, bayi berusia 3 bulan diberikan imunisasi DPT 2 dan Polio 3, untuk bayi berusia 4 bulan diberikan imunisasi DPT 3, Polio 4, dan IPV, dan untuk bayi berusia 9 bulan diberikan imunisasi Campak.

Imunisasi Hepatitis B diberikan untuk mencegah penyakit Hepatitis B yang dapat menyebabkan pengerasan hati yang berujung pada kegagalan fungsi hati dan kanker hati. Imunisasi BCG diberikan guna mencegah penyakit tuberculosis. Imunisasi Polio diberika 4 kali pada usia 1 bulan, 2 bulan, 3 bulan dan 4 bulan, untuk mencegah lumpuh layu. Imunisasi IPV diberikan dengan tujuan mencegah virus polio. Imunisasi campak diberikan untuk mencegah penyakit campak yang dapat mengakibatkan radang paru berat (pneumonia), diare atau menyerang otak.

~Halaman ini sengaja dikosongkan~