BAB II

LANDASAN TEORI

2.1 *Rubella*

Rubella adalah salah satu penyakit umum yang menyebar di seluruh dunia dan menyerang berbagai umur dengan gejala yang bervariasi (Bachtiar, Raksanagara, Widhiastuti, Judistiani, & Hardiana, 2015).

Rubella pada anak sering hanya menimbulkan gejala demam ringan atau bahkan tanpa gejala sehingga sering tidak terlaporkan, sedangkan *rubella* pada wanita dewasa sering menimbulkan sakit sendi (*arthritis* atau *arthralgia)*. *Rubella* pada wanita hamil terutama pada kehamilan trimester pertama dapat mengakibatkan keguguran atau bayi lahir dengan cacat bawaan yang disebut *congenital rubella syndrome (*CRS) (Kemenkes RI, 2018).

Congenital Rubella Syndrome (CRS) adalah suatu kumpulan gejala penyakit terdiri dari katarak (kekeruhan lensa mata), penyakit jantung bawaan, gangguan pendengaran, dan keterlambatan perkembangan, termasuk keterlambatan bicara dan disabilitas intelektual.
Sindrom *rubella* kongenital disebabkan infeksi virus *rubella* pada janin selama masa kehamilan akibat ibu tidak mempunyai kekebalan terhadap virus *rubella*. Seorang anak dapat menunjukkan satu atau lebih gejala CRS dengan gejala tersering adalah gangguan pendengaran (Fitriany & Husna, 2018).

Risiko bayi yang terkena CRS dan keparahan dari cacat lahir tergantung di usia kehamilan ketika ibu terkena *rubella*:

1. Infeksi pada awal kehamilan paling berbahaya (<12 minggu).
    a. Minggu 1 – 10 = 90% terkena CRS.
    b. Minggu 11 – 12 = 33% terkena CRS.
    c. Minggu 13 – 14 = 11% terkena CRS.
    d. Minggu 15 – 16 = 24% terkena CRS.
    e. Minggu ≥ 17 = 0% terkena CRS.
2. Menyebabkan lahir mati atau prematur atau *abortus*.

3. Manifestasi klinis atau organ yang terserang tergantung usia kehamilan saat infeksi. (Pencegahan, Measles, & Rubela, n.d.)

Setiap tahun melalui kegiatan surveilans dilaporkan lebih dari 11.000 kasus suspek campak, dan hasil konfirmasi laboratorium menunjukkan 12–39% di antaranya adalah campak pasti (*lab confirmed*) sedangkan 16–43% adalah *rubella* pasti. Dari tahun 2010 sampai 2015, diperkirakan terdapat 23.164 kasus campak dan 30.463 kasus *rubella*. Jumlah kasus ini diperkirakan masih lebih rendah dibanding angka sebenarnya di lapangan, mengingat masih banyaknya kasus yang tidak terlaporkan, terutama dari pelayanan kesehatan swasta serta kelengkapan laporan *surveilans* yang masih rendah.

Di Indonesia, *Rubella* merupakah salah satu masalah kesehatan masyarakat yang memerlukan upaya pencegahan efektif. Data *surveilans* selama lima tahun terakhir menunjukan 70% kasus *rubella* terjadi pada kelompok usia <15 tahun. Selain itu, berdasarkan studi tentang estimasi beban penyakit CRS di Indonesia pada tahun 2013 diperkirakan terdapat 2.767 kasus CRS, 82/100.000 terjadi pada usia ibu 15-19 tahun dan menurun menjadi 47/100.000 pada usia ibu 40-44 tahun (Kemenkes RI, 2018).

2.1.1 Gejala-Gejala *Rubella*

2.1.1.1 Masa *Prodromal*

1. Masa *prodromal* 1-5 hari.
2. Ditandai dengan *malaise, anoreksia, limfadenopati*, disusul demam *subfebris*, *konjungtivitis ringan, koriza*, nyeri tenggorokan dan batuk.
3. Gejala cepat menurun setelah hari pertama timbulnya ruam.
4. Demam berkisar 38˚C – 38,7˚C (*subfebril*). Biasanya timbul dan menghilang bersamaan dengan ruam kulit.
   (Pencegahan et al., n.d.)

2.1.1.2 Karakteristik Erupsi Kulit

1. *Makulopapular, eritematosa, diskret*.

2. Pertama di muka dan menyebar ke bawah dengan cepat (leher, badan, dan ekstremitas).
3. Saat ruam tampak lebih jelas di ekstremitas, di tempat lain mulai menghilang.
4. *Enantema* pada *rubella (forschheimer spots)* berupa bercak *pinpoint* atau lebih besar, warna merah muda, tampak pada *palatum mole* sampai *uvula*.
5. Bercak *Forschheimer* bukan tanda *patognomonik*.
   (Pencegahan et al., n.d.)

2.1.1.3 Tanda *Patognomonic*

1. Ruam muncul bersamaan dengan demam yang tidak tinggi (*subfebril*).
2. Adanya *limfadenopati* khususnya pada daerah belakang telinga dan *oksipital* sangat menunjang diagnosis,beberapa hari sebelum demam dan ruam.
   (Pencegahan et al., n.d.)

2.1.2 Penyebab *Rubella*

Sindrom *rubella* kongenital disebabkan infeksi virus *rubella* pada janin selama masa kehamilan akibat ibu tidak mempunyai kekebalan terhadap virus *rubella*. Virus *rubella* ditransmisikan melalui pernapasan yaitu melalui *droplet* yang dikeluarkan oleh seseorang yang terinfeksi *rubella*, setelah terkena *droplet*, virus ini akan mengalami replikasi di *nasofaring* dan di daerah kelenjar getah bening. *Viremia* terjadi antara hari ke-5 sampai hari ke-7 setelah terpajan virus *rubella*. Infeksi *rubella* menyebabkan kerusakan janin karena proses pembelahan terhambat. Diagnosis dari CRS bisa ditegakkan melalui *anamnesis*, pemeriksaan fisik dan pemeriksaan pebunjang. Pemeriksaan laboratorium untuk menunjang diagnosis CRS antara lain: isolasi virus, pemeriksaan *serologik* (ELISA) dan pemeriksaan terhadap RNA virus *rubella.* Terapi untuk CRS sendiri hanya bersifat suportif untuk defek - defek yang dialami. Penting untuk mencegah CRS adalah dengan vaksin MMR sebelum hamil. Prognosis untuk CRS lebih buruk dibandingkan dengan *rubella postnatal* karena disertai kerusakan organ *multiple* yang berat (Fitriany & Husna, 2018).

2.1.3 Langkah Pencegahan *Rubella*

Tidak ada pengobatan untuk penyakit *rubella* namun penyakit ini dapat di cegah. Imunisasi dengan vaksin MR adalah pencegahan terbaik karena satu vaksin bisa mencegah dua penyakit sekaligus yaitu campak dan *rubella* (*buku_petunjuk_untuk_guru_dan_kader.pdf*, n.d.).

Pencegahan *rubella* yang paling efektif adalah dengan vaksinasi. Selain vaksin, mencegah penularan dan penyebaran *rubella* juga penting. Cara-caranya meliputi:

1. Berikan anak minuman dingin.
2. Kenakan ia pakaian tipis.
3. Berikan parasetamol khusus bayi (cek usia yang dianjurkan di kemasan) untuk menurunkan suhu tubuhnya.
4. Bagi anak-anak balita, pada usia 15 bulan atau 12 bulan, jika ia tidak mendapatkan imunisasi campak, berikan vaksinasi *Mumps Measles Rubela* (MMR) untuk mencegah risiko tinggi yang membahayakan bagi kesehatan dengan mendatangi layanan kesehatan terdekat.
   (Pengembangan Sistem et al., 2013)

2.1.4 Vaksin MR

Vaksin MR adalah kombinasi vaksin Campak/*Measles* (M) dan *Rubella* (R) untuk perlindungan terhadap penyakit campak dan *rubella* (*buku_petunjuk_untuk_guru_dan_kader.pdf*, n.d.).

Imunisasi MR *(Measles, Rubella)* merupakan imunisasi yang di gunakan dalam memberikan kekebalan terhadap penyakit campak (*measles*) dan campak jerman (*rubella*). Dalam imunisasai MR *(Measles, Rubella)*, antigen yang di pakai adalah virus campak *strain Edmonson* yang dilemahkan, virus *rubella* strai RA 27/3, dan virus gondog (Hidayat, 2008) dalam (Studi, Pendidik, Diploma, & Kesehatan, 2017).

Vaksin MR hanya di berikan pada bayi usia 9 bulan sampai dengan kurang dari 15 tahun. Dan vaksin MR ini tidak boleh di berikan pada wanita hamil.

Karena berdasarkan data *surveilans* kasus terbanyak di Indonesia adalah pada usia di bawah 15 tahun. Untuk mencapai eliminasi *rubella* di Indonesia, maka kelompok usia inilah yang menjadi fokus utama dan harus segera di imunisasi (*20-FAQ-Fakta-MR.pdf*, n.d.).

2.1.4.1 Sebagian Orang yang Tidak Boleh Menerima Vaksin MR

1. Memiliki alergi berat yang dapat mengancam jiwa.
2. Sedang hamil.
3. Memiliki sistem kekebalan tubuh yang lemah.
4. Memiliki orangtua, saudara laki-laki atau perempuan dengan riwayat masalah sistem kekebalan tubuh.
5. Pernah mengalami kondisi yang membuat mereka mudah mengalami lebam atau pendarahan.
6. Baru saja menjalani transfusi darah atau menerim produk darah lainnya.
7. Menderita *tuberkulosis*.
8. Sudah mendapat vaksin lainnya dalam 4 minggu terakhir.
9. Sedang merasa tidak sehat.

   (Vaksin, n.d.)

2.1.4.2 Risiko Reaksi Vaksin

1. Kejadian ringan, sebagai berikut.
   a. Nyeri pada lengan akibat injeksi.
   b. Demam.
   c. Kemerahan atau ruam di lokasi injeksi.
   d. Pembengkakan kelenjar di pipi atau leher.
2. Kejadian sedang, sebagai berikut.
   a. Kejang (tersentak atau terbelalak) seringkali berhubungan dengan demam.
   b. Nyeri dan kaku pada persendian yang bersifat sementara, kebanyakan di alami remaja atau wanita dewasa.
   c. Jumlah *trombosit* rendah yang bersifat sementara, yang dapat menyebabkan pendarahan atau lebam yang tidak lazim.

d. Ruam di sekujur tubuh.

3. Kejadian berat (sangat jarang terjadi), sebagai berikut.
   a. Ketulian.
   b. Kejang yang berlangsung lama, koma atau penurunan keasadaran.
   c. Kerusakan otak.

   (Vaksin, n.d.)

2.2 Sistem Pakar

2.2.1 Definisi Sistem Pakar

Secara umum, sistem pakar adalah sistem yang berusaha mengadopsi pengetahuan manusia ke komputer yang dirancang untuk memodelkan kemampuan menyelesaikan masalah seperti layaknya seorang pakar. Dengan sistem pakar ini, orang awam pun dapat menyelesaikan masalahnya atau hanya sekedar mencari suatu informasi berkualitas yang sebenarnya hanya dapat diperoleh dengan bantuan para ahli di bidangnya. Sistem pakar ini juga akan dapat membantu aktivitas para pakar sebagai asisten yang berpengalaman dan mempunyai pengetahuan yang dibutuhkan. Dalam penyusunannya, sistem pakar mengkombinasikan kaidah-kaidah penarikan kesimpulan (*inference rules*) dengan basis pengetahuan tertentu yang diberikan oleh satu atau lebih pakar dalam bidang tertentu. Kombinasi dari kedua hal tersebut disimpan dalam komputer, yang selanjutnya digunakan dalam proses pengambilan keputusan untuk penyelesaian masalah tertentu (Widyawati, 2018).

Sistem pakar merupakan program komputer untuk dapat meniru proses pemikiran dan pengetahuan pakar untuk menyelesaikan suatu masalah yang spesifik. Implementasi sistem pakar banyak digunakan untuk kepentingan masyarakat karena sistem pakar dipandang sebagai cara penyimpanan pengetahuan pakar dalam bidang tertentu ke dalam suatu program, sehingga dapat memberikan keputusan dan melakukan penalaran secara cerdas (Putri & Mustafidah, 2011).

Sistem pakar merupakan suatu sistem yang berbasis pengetahuan *(knowledge-based system)*, yaitu menggunakan pengetahuan manusia yang

disimpan manusia dalam *database* untuk memecahkan permasalahan yang biasanya memerlukan keahlian manusia (Turnip Mardi, 2015:1.) dalam (Suryana & Sallaby, 2019).

2.2.2 Jenis – Jenis Sistem Pakar

Adapun beberapa jenis – jenis sistem pakar beserta kegunaannya yang dapat dilihat dalam tabel 2.1 berikut.

Tabel 2.1 Jenis – jenis sistem pakar dan kegunaannya (Ismail, 2017)

Sistem Pakar	Kegunaan
MYCIN	Diagnosa Penyakit
DENDRAL	Mengidentifikasi struktur molecular campuran yang tak dikenal
XCON & XSEL	Membantu konfigurasi sistem komputer besar
SOPHIE	Analisis sirkit elektronik
Prospector	Digunakan di dalam geologi untuk membantu mencari dan menemukan deposit

2.2.3 Ciri – Ciri Sistem Pakar

Disebabkan oleh keheuristikannya dan sifatnya yang berdasarkan pada pengetahuan sehingga umumnya sistem pakar mempunyai ciri-ciri sebagai berikut:

1. Terbatas pada domain keahlian tertentu.
2. Berdasarkan pada kaidah/rule tertentu.
3. Dapat digunakan dalam berbagai jenis komputer.
4. Mudah dimodifikasi, yaitu dengan menambah atau menghapus suatu kemampuan dari basis pengetahuannya.
5. Sistem dapat mengaktifkan kaidah secara searah yang sesuai, dituntun oleh dialog dengan pemakai.
6. Bekerja secara sistematis berdasarkan pengetahuan dan mekanisme tertentu.
7. Pengambilan keputusan berdasarkan kaidah-kaidah tertentu dan dapat merespons masukan user (melalui kotak dialog).

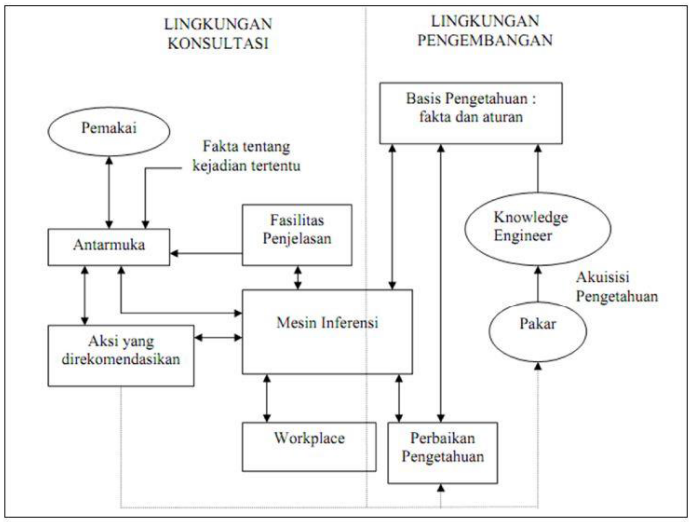
8. Dapat menalar data-data yang tidak pasti dan memberikan beberapa alasan pemilihan.
9. Dikembangkan secara bertahap dan terbatas pada bidang keahlian tertentu saja.
10. *Output*nya berupa saran atau anjuran.
11. *Knowledge base* dan *inference engine* terpisah.

    (Juniawan, 2017)

2.2.4 Struktur Sistem Pakar

Sistem pakar disusun oleh dua bagian utama yaitu lingkungan pengembangan (*development environtment*) dan lingkungan konsultasi (*consultant environtment*). Lingkungan pengembangan sistem pakar digunakan untuk memasukan pengetahuan pakar ke dalam lingkungan sistem pakar, sedangkan lingkungan konsultasi digunakan oleh penggunan yang bukan pakar guna memperoleh pengetahuan pakar (Lestari, Diantoro, & Komputer, 2018).

Komponen – komponen sistem pakar dalam kedua bagian tersebut dapat dilihat dalam gambar 2.1 berikut.



Gambar 2.1 Struktur Sistem Pakar (Ismail, 2017)

2.2.5 Kelebihan Sistem Pakar

Adapun keuntungan dari penggunaan sistem pakar antara lain:

1. Memungkinkan pengguna yang bukan seorang pakar pada bidang tertentu dapat mengerjakan tugas dari seorang pakar.
2. Bisa melakukan proses yang sama secara berulang.
3. Sistem pakar dapat menyimpan pengetahuan dan keahlian dari pakar.
4. Dengan adanya sistem pakar produktivitas dan *output* sistem dapat di tingkatkan.
5. Meningkatkan kualitas.
6. Mampu mengambil dan melestarikan keahlian para pakar.
7. Mampu beroperasi dalam lingkungan yang berbahaya.
8. Memiliki kemampuan untuk mengakses pengetahuan.
9. Memiliki reabilitas.
10. Meningkatkan kapabilitas sistem komputer.
11. Memiliki kemampuan untuk bekerja dengan informasi yang tidak lengkap dan mengandung ketidakpastian.
12. Sebagai media pelengkap dalam pelatihan.
13. Meningkatkan kapabilitas dalam penyelesaian masalah.
14. Menghemat waktu dalam pengambilan keputusan.

(Andriani, 2017)

2.2.6 Kelemahan Sistem Pakar

Selain memiliki kelebihan, sistem pakar juga memiliki kelemahan, yaitu sebagai berikut:

1. Biaya yang di perlukan untuk membuat, memelihara, dan mengembangkan sistem pakar sangat mahal.
2. Sulit di kembangkan, karena ketersediaan pakar di bidangnya dan kepakaran sulit di ekstrak dari manusia karena terkadang sulit bagi seorang pakar untuk menjelaskan langkah mereka dalam menangani masalah.
3. Sistem pakar tidak 100% benar karena seseorang yang terlibat dalam pembuatan sistem pakar tidak selalu benar. Oleh karena itu setelah pembuatan sistem pakar harus di lakukan pengujian terlebih dahulu secara teliti sebelum di gunakan.

4. Pendekatan oleh setiap pakar untuk suatu situasi atau *problem* bisa berbeda – beda, meskipun sama – sama benar.
5. *Transfer* pengetahuan dapat bersifat subjektif dan bias.
6. Kurangnya rasa percaya pengguna dapat menghalangi pemakaian sistem pakar.

(Andriani, 2017)

2.3 Metode *Forward Chaining*

2.3.1 Definisi *Forward Chaining*

Forward chaining merupakan cara penalaran dengan memulai dari fakta terlebih dahulu untuk menguji kebenaran hipotesis atau mencocokkan fakta atau pernyataan di mulai dari bagian sebelah kiri dulu (*IF* dulu). *Forward Chaining* merupakan grup dari *multiple* inferensi yang melakukan pencarian dari suatu masalah kepada solusinya. Jika klausa premis sesuai dengan situasi (bernilai *TRUE*), maka proses akan meng-*assert* konklusi. *Forward chaining* cocok di gunakan untuk suatu aplikasi yang menghasilkan *tree* yang lebar dan tidak dalam (Andriani, 2017).

Pada metode *forward chaining*, pencarian dapat di lakukan dengan dua cara, yaitu:

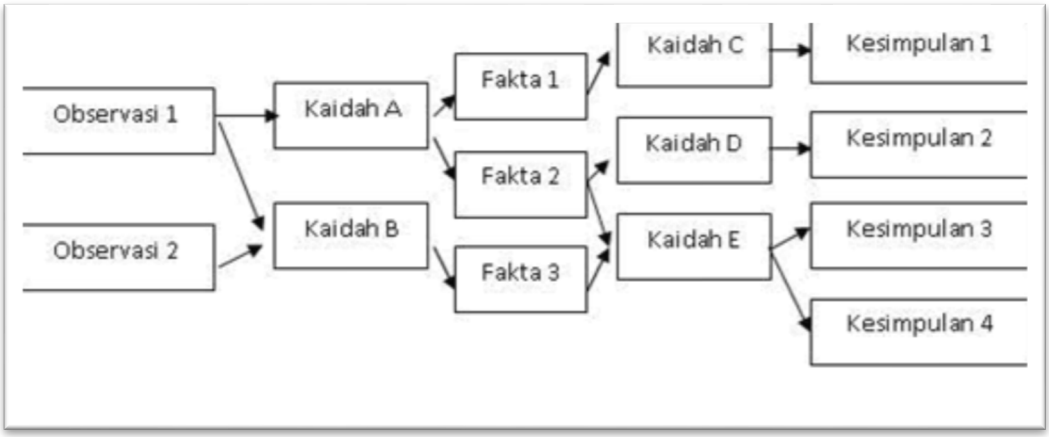
1. Meng*input*kan semua data ke dalam sistem pakar dalam sesi konsultasi. Cara seperti ini tepat dan berguna pada sistem pakar dimana proses di dalamnya terotomastisasi dan langsung menerima data dari *database* atau dari satu *set* sensor.
2. Memberikan elemen spesifik dari data yang di peroleh selama sesi konsultasi dalam sistem pakar. Cara ini mengurangi jumlah data yang di minta, sehingga data yang di minta hanya data yang benar – benar di butuhkan oleh sistem pakar tersebut yang nantinya akan di gunakan untuk mengambil keputusan. (Andriani, 2017)

2.3.2 Konsep Dasar Metode *Forward Chaining*

Berikut merupakan langkah – langkah dalam membuat sistem pakar dengan menggunakan metode *forward chaining*, yaitu:

a. Pendefenisian masalah dimulai dengan pemilihan domain masalah dan akuisi pengatahuan.
b. Pendefenisian data *input* untuk memulai inferensi karena diperlukan oleh sistem *forward chaining.*
c. Pendefenisian struktur pengendalian data untuk membantu mengendalikan pengaktifan suatu aturan.
d. Penulisan kode awal dalam domain pengatahuan.
e. Pengujian sistem agar dapat mengatahui sejauh mana sistem berjalan.
f. Perancangan antarmuka dengan basis pengatahuan.
g. Pengembangan sistem.
h. Evaluasi sistem.

(Jamkhandi & Disouza, 2012)



Gambar 2.2 Konsep Dasar Metode *Forward Chaining* (Jamkhandi & Disouza, 2012)

2.3.3 Kelebihan Metode *Forward Chaining*

Adapun kelebihan dari metode *forward chaining*, yaitu sebagai berikut.

1. Metode ini akan bekerja dengan baik ketika *problem* bermula dari mengumpulkan atau menyatukan informasi lalu kemudian mencari kesimpulan apa yang dapat di ambil dari informasi tersebut.
2. Metode ini mampu menyediakan banyak sekali informasi dari hanya sejumlah kecil data.
3. Merupakan pendekatan paling sempurna untuk beberapa tipe dari *problem solving tasks*, yaitu *planning*, *monitoring*, *control*, dan *interpretation*.
   (Dwi, Supartha, Sari, & Informatika, 2014)

2.3.4 Kelemahan Metode *Forward Chaining*

Selain memiliki kelebihan metode *forward chaining* juga memiliki kelemahan, sebagai berikut.

1. Tidak dapat mengenali dimana beberapa fakta lebih penting dari fakta lainnya.
2. Sistem dapat menanyakan pertanyaan yang tidak berhubungan, meskipun jawaban dari pertanyaan tersebut penting. Hal ini dapat membingungkan *user* dalam menjawab pertanyaan pada subyek yang tidak berhubungan.
   (Dwi et al., 2014)

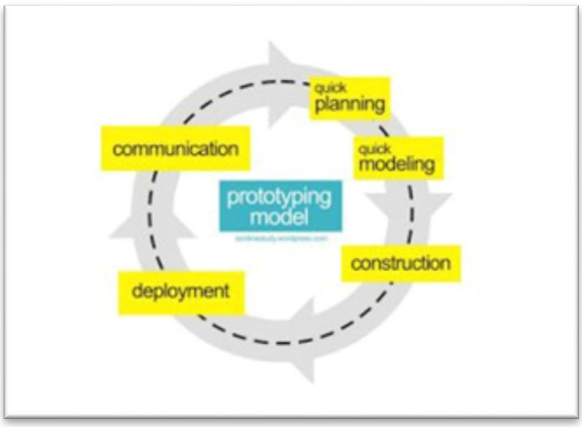
2.4 Pengembangan Sistem

2.4.1 *Prototype*

Prototyping pardigma dimulai dengan pengumpulan kebutuhan, pengembang bertemu dengan pengguna dan mengidentifikasikan objektif keseluruhan dari perangkat lunak, selanjutnya mengidentifikasi segala kebutuhan yang diketahui secara garis besar di mana definisi-definisi lebih jauh merupakan keharusan, kemudian dilakukan perancangan kilat, lalu diakhiri dengan evaluasi *prototyping* (Sari & Komputer, 2016).

Prototype dimulai dengan mengumpulkan kebutuhan yang akan di rancang. Pengembang mendefinisikan *object* keseluruhan dari perangkat lunak, mengidentifikasi segala aktifitas yang diketahui dan kemudian melakukan "perancangan kilat". Perancangan kilat berfokus pada penyajian dari aspek-aspek

perangkat lunak tersebut yang akan nampak bagi pelanggan atau pemakai (contohnya pendekatan *input* dan format *output*) (Puspita, Ali, Kasus, & Informatika, 2019), prototype memiliki 5 tahapan seperti pada gambar 2.3 berikut:



Gambar 2.3 Model Pengembangan *Prototype* (Susanto & Andriana, 2016)

2.4.1.1 Tujuan Prototype

Dibuatnya sebuah *Prototyping* bagi pengembang sistem bertujuan untuk mengumpulkan informasi dari pengguna sehingga pengguna dapat berinteraksi dengan model *prototype* yang dikembangkan, sebab *prototype* menggambarkan versi awal dari sistem untuk kelanjutan sistem sesungguhnya yang lebih besar (Purnomo, 2017).

Tujuan lainnya dari penggunaan *prototyping* adalah :

1. Mewujudkan sistem sesungguhnya dalam sebuah replika sistem yang akan berjalan, menampung masukan dari pengguna untuk kesempurnaan sistem.
2. Pengguna akan lebih siap menerima setiap perubahan sistem yang berkembang sesuai dengan berjalannya *prototype* sampai dengan hasil akhir pengembangan yang akan berjalan nantinya.
3. *Prototype* dapat ditambah maupun dikurangi sesuai berjalannya proses pengembangan. Kemajuan tahap demi tahap dapat diikuti langsung oleh pengguna.
4. Penghematan sumberdaya dan waktu dalam menghasilkan produk yang lebih baik dan tepat guna bagi pengguna.

   (Purnomo, 2017)

2.4.1.2 Tahap-tahap *Prototype*

1. *Requirement analysis and definition*

   Pada tahap ini dilakukan analisis terhadap masalah apa yang sedang terjadi pada objek penelitian. Analisis permasalahan dilakukan dengan studi literatur dan wawancara. Selain melakukan analisis permasalahan, dilakukan juga analisis kebutuhan, analisis kebutuhan ini nantinya dijadikan sebagai alat bantu yang digunakan dalam proses pembuatan *prototype* hingga menjadi aplikasi final.

2. *User interface prototyping*

   Setelah analisis kebutuhan sistem telah dilakukan, pada tahap ini dilakukan identifikasi kembali kebutuhan sistem tersebut. Apabila kebutuhan sistem telah teridentifikasi dengan baik, dapat dilakukan proses selanjutnya yaitu pembuatan *user interface prototype*. *User interface prototype* ini adalah tampilan dan interaksi tentang aplikasi yang dibangun.

3. *Architecture & component design and prototyping*

   Setelah *prototype user interface* telah selesai dibuat, proses selanjutnya adalah membuat desain dan *prototype* arsitektur dan komponen-komponen aplikasi yang dibangun dan nantinya digunakan sebagai acuan untuk membuat aplikasi *final*.

4. *Implementation and system testing*

   Selanjutnya, jika seluruh proses sebelumnya telah dilakukan maka dihasilkan *prototype* yang digunakan sebagai acuan dalam pembuatan aplikasi. Setelah aplikasi selesai dibangun, maka dilakukan proses pengujian atau *testing* aplikasi untuk menguji atau mengetahui kualitas dari aplikasi yang telah dibangun. Setelah dilakukan pengujian terhadap aplikasi tersebut, pihak perusahaan berhak untuk melakukan evaluasi terhadap aplikasi tersebut, apakah aplikasi tersebut sesuai dengan kebutuhan perusahaan atau tidak. Apabila aplikasi sudah sesuai dengan kebutuhan perusahaan, maka aplikasi siap untuk diimplementasikan pada perusahaan tersebut.

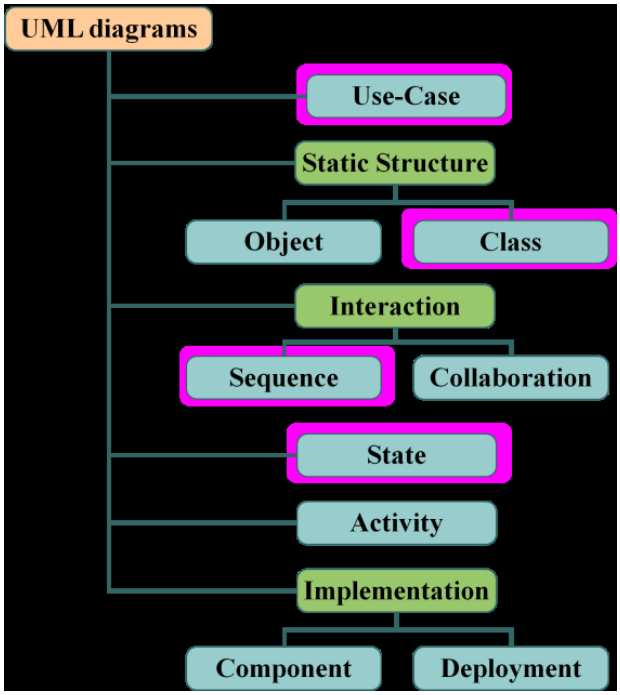
   (Pradipta, Yuli Adam Prasetyo, ST., & Nia Ambarsari, S.Si., 2015)

2.4.2 *Unified Modelling Language* (*UML)*

Menurut Nugroho (2010: 6), UML adalah bahasa pemodelan untuk sistem atau perangkat lunak yang berparadigma berorientasi objek (Bangun et al., 2014).

UML merupakan kesatuan dari bahasa pemodelan yang dikembangkan oleh Booch, *Object Modeling Technique* (OMT) dan *Object Oriented Software Engineering* (OOSE). Metode *Booch* dari Grady Booch sangat terkenal dengan nama Metode *Design Object Oriented* (Kamda, 2012).

UML sendiri juga memberikan standar penulisan kelas-kelas dalam bahasa program yang spesifik, skema *database*, dan komponen-komponen yang diperlukan dalam sistem *software*. *UML* sebagai sebuah bahasa yang memberikan tatanan penulisan kata-kata dalam '*MS Word'* untuk kegunaan komunikasi. Sebuah bahasa model adalah sebuah bahasa yang mempunyai konsep aturan penulisan serta secara fisik mempresentasikan dari sebuah sistem (Berbasis & Delphi, 2015).



Gambar 2.4 Diagram UML (Studi, Komputer, & Mulawarman, 2011)

2.4.2.1 Tujuan Pemanfaatan UML

Menurut Sugrue J. (2009) dalam (Studi et al., 2011), tujuan utama UML adalah sebagai berikut.

1. Menyediakan bagi pengguna (analisis dan desain sistem) suatu bahasa pemodelan *visual* yang ekspresif sehingga mereka dapat mengembangkan dan melakukan pertukaran model data yang bermakna.
2. Menyediakan mekanisme yang spesialisasi untuk memperluas konsep inti.
3. Karena merupakan bahasa pemodelan *visual* dalam proses pembangunannya maka UML bersifat *independent* terhadap bahasa pemrograman tertentu.
4. Memberikan dasar formal untuk pemahaman bahasa pemodelan.
5. Mendorong pertumbuhan pasar terhadap penggunaan alat desain sistem yang berorientasi objek (OO).
6. Mendukung konsep pembangunan tingkat yang lebih tinggi seperti kolaborasi, kerangka, pola dan komponen terhadap suatu sistem.
7. Memiliki integrasi praktik terbaik.

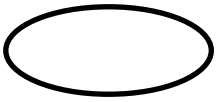
   (Studi et al., 2011)

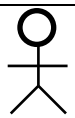
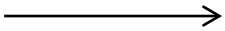
2.4.2.2 *Usecase Diagram*

Use Case Diagram adalah gambar dari beberapa atau seluruh aktor dan *use case* dengan tujuan yang mengenali interaksi mereka dalam suatu sistem. *Use case diagram* menggambarkan fungsionalitas yang diharapkan dari sebuah sistem, yang ditekankan adalah "apa" yang diperbuat sistem, dan bukan "bagaimana". Sebuah *use case* mempresentasikan sebuah interaksi antara *actor* dan sistem (Purwati & Rahardi, 2018).

Seorang atau sebuah aktor adalah sebuah entitas manusia atau mesin yang berinteraksi dengan sistem untuk melakukan pekerjaan-pekerjaan tertentu (Dharwiyanti, 2003).

Tabel 2.2 Diagram *Use Case*

Gambar	Keterangan
	*Use case* menggambarkan fungsionalitas yang di sediakan sistem sebagai unit-unit yang bertukar pesan

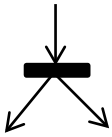
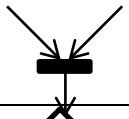
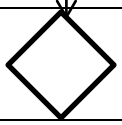
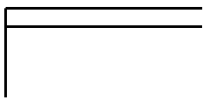
	antar unit dengan aktor, yang di nyatakan dengan menggunakan kata kerja.
	*Actor* adalah *abstraction* dari orang atau sistem yang lain yang mengaktifkan fungsi dari target sistem.
	Asosiasi antara aktor dan *use case*, di gambarkan dengan garis tanpa panah yang mengindikasikan siapa atau apa yang meminta interaksi secara langsung dan bukannya mengindikasikan data.
	Asosiasi antara aktor dan *use case* yang menggunakan panah terbuka untuk mengindikasikan bila aktor berinteraksi sacara pasif dengan sistem.
<<include>	*Include*, merupakan di dalam *use case* lain (*required*) atau pemanggilan *use case* oleh *use case* lain.
 <<extends>>	*Extend*, merupakan perluasan dari *use case* lain jika kondisi atau syarat terpenuhi.

2.4.2.3 *Activity Diagram*

Activity diagram menggambarkan rangkaian aliran dari aktifitas, digunakan untuk mendeskripsikan aktivitas yang dibentuk dalam suatu operasi sehingga dapat juga digunakan untuk aktifitas lainnya seperti *use case* atau interaksi. *Activity Diagram* berupa *flowchart* yang digunakan untuk memperlihatkan aliran kerja dari sistem (Purwati & Rahardi, 2018).

Activity diagram merupakan *state diagrams* khusus, di mana sebagian besar *state* adalah *action* dan sebagian besar transisi di-*trigger* oleh selesainya *state* sebelumnya (*internal processing*). Oleh karena itu *activity diagrams* tidak menggambarkan *behaviour internal* sebuah sistem (dan interaksi antar subsistem) secara eksak, tetapi lebih menggambarkan proses-proses dan jalur-jalur aktivitas dari level atas secara umum (Dharwiyanti, 2003).

Tabel 2.3 Diagram Aktivitas

Gambar	Keterangan
	*Start point*, di letakkan pada pojok kiri atas dan merupakan awal aktivitas.
	*End point*, merupakan akhir aktivitas.
	*Activities*, menggambarkan suatu proses atau kegiatan bisnis.
	*Fork* atau percabangan, di gunakan untuk menujukkan kegiatan yang di lakukan secara paralel atau untuk menggabungkan dua kegiatan paralel menjadi satu.
	*Join* (penggabungan) atau *rake*, di gunakan untuk menujukkan adanya dekomposisi.
	*Decision points*, menggambarkan pilihan untuk pengambilan keputusan, *true* atau *false*.
	*Swimlane*, pembagian *activity diagram* untuk menujukkan siapa dan melakukan apa.

2.4.2.4 *Class Diagram*

Class diagram menggambarkan struktur data dan deskripsi *class, package* dan objek beserta hubungan satu sama lain. *Class diagram* berfungsi untuk menjelaskan tipe dari objek sistem dan hubungannya dengan objek yang lain (Purwati & Rahardi, 2018).

Class memiliki tiga area pokok, yaitu :

1. Nama (dan *stereotype*).
2. Atribut.
3. Metoda.

(Dharwiyanti, 2003)

Tabel 2.4 Diagram kelas

Multiplicity	Penjelasan
1	Satu dan hanya satu.
0..*	Boleh tidak ada atau 1 atau lebih.
1..*	1 atau lebih.

0..1	Boleh tidak ada, maksimal 1.
n..n	Batasan antara. Contoh 2..4 mempunyai arti minimal 2 maksimal 4.

2.5 Bahasa Pemrograman

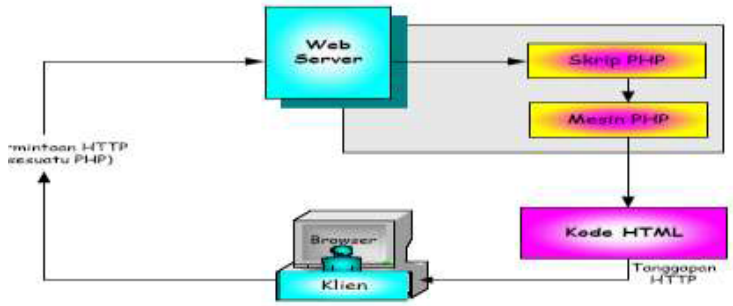
2.5.1 PHP *(Hypertext Preprocessor)*

Merupakan suatu *script* yang bersifat yang digunakan untuk membuat sebuah *web* menjadi lebih menarik, dinamis, dan interaktif. Dengan PHP kita dapat mengolah data yang diambil dengan sebuah *form*, membuat aplikasi-aplikasi tertentu dalam sebuah *web*, ataupun membuat *database* dalam sebuah *web*. (Betha, 2001:36) dalam (S. Handayani & Kanedi, 2014).

Fungsi utama PHP dalam membangun *website* adalah untuk melakukan pengolahan data pada *database*. Data *website* akan dimasukkan ke *database*, diedit, dihapus, dan ditampilkan pada *website* yang diatur oleh PHP (No, Josi, & Josi, 2017).

Sistem kerja dari PHP diawali dengan permintaan yang beasal dari halaman *website* oleh *browser*. Berdasarkan URL atau alamat *website* dalam jaringan internet, *browser* akan menemukan sebuah alamat dari *webserver*, mengidentifikasi halaman yang dikehendaki, dan menyampaikan segala informasi yang dibutuhkan oleh *webserver*. Selanjutnya *webserver* akan mencarikan berkas yang diminta dan menampilkan isinya di *browser*. *Browser* yang mendapatkan isinya segera menerjemahkan kode HTML dan menampilkannya. Jika yang dipanggil oleh *user* adalah halaman yang mengandung *script* PHP, maka prinsipnya sama dengan memanggil kode HTML, namun pada saat permintaan dikirim ke *web-server*, *web-server* akan memeriksa tipe *file* yang diminta *user*. Jika tipe *file* yang diminta adalah PHP, maka akan memeriksa isi *script* dari halaman PHP tersebut. Apabila dalam *file* tersebut tidak mengandung *script* PHP, permintaan *user* akan langsung ditampilkan ke *browser*, namun jika dalam *file* tersebut mengandung *script* PHP, maka proses akan dilanjutkan ke modul PHP sebagai mesin yang menerjemahkan *script-script* PHP dan mengolah *script*

tersebut, sehingga dapat dikonversikan ke kode-kode HTML lalu ditampilkan ke *browser user* (Firman et al., 2016).



Gambar 2.5 Konsep Kerja PHP (Computech, 2008)

Sistem *database* yang di dukung oleh PHP, yaitu *Oracle, Sysbase*, mSQL, MySQL, *Solid, Generic* ODBC, *Postgres* SQL (Computech, 2008).

2.5.1.1 *Script* PHP

Script PHP diawali dengan *tag* (<?) dan diakhiri dengan *tag* (?>). Setiap baris perintah atau *statement* harus diakhiri dengan menggunakan tanda titik koma (;) (Ii, 2005).

Cara penulisannya di bedakan menjadi 2, yaitu sebagai berikut (PHP dan MySQL, Didik Dwi Prastyo, 2003, h : 4-5) dalam (Ii, 2005).

a. *Embedded script*

   Script yang dari *embedded script* adalah *script* PHP yang disisipkan di antara *tag-tag* HTML.

```
<html>
<head>
<title> Contoh Embedded Script PHP</title>
</head>
<body>
<?php
Echo”Halo, Selamat Datang Di Pemrograman PHP”;
?>
```

Gambar 2.6 *Embedded Script* (Ii, 2005)

b. *Non embedded script*

Script PHP pada cara ini digunakan sebagai murni pembuatan program PHP, *tag* HTML yang dihasilkan untuk membuat dokumen merupakan bagian dari *script* PHP.

```
<?php
echo”<html>”;
echo”<head>”;
echo”<title>”non embedded script”</title>”;
echo”</head>”;
echo”<body>”;
echo””;
echo”</body>”;
echo”</html>”;
?>
```

Gambar 2.7 *Non Embedded Script* (Ii, 2005)

2.5.1.2 Kelebihan PHP

PHP memiliki kelebihan-kelebihan sebagai berikut.

1. *Web* menggunakan PHP dapat dengan mudah dibuat dan memiliki kecepatan akses yang cukup tinggi.
2. Skrip-skrip PHP dapat berjalan dalam *web server* yang berbeda dan dalam *system* operasi yang berbeda pula. PHP dapat berjalan disistem operasi UNIX, *windows* dan *macintosh*.
3. PHP diterbitkan secara gratis.
4. PHP juga dapat berjalan pada *web server Microsoft Personal Web Server, Apache*, IIS, *Xitami* dan sebagainya.
5. PHP adalah termasuk bahasa *embedded* (bisa ditempel atau diletakan dalam *tag* HTML).
6. PHP termasuk *server side programming*.

(Computech, 2008)

2.5.2 MySQL

MySQL merupakan sebuah *Relational Database Management System* (RDBMS) yang bersifat *open source* (No et al., 2017).

Mysql adalah salah satu jenis *database server* yang sangat terkenal. Kepopulerannya disebabkan MySQL menggunakan SQL sebagai bahasa dasar untuk mengakses *database*nya. MySQL bersifat *free* dengan lisensi GNU *General Public License* (GPL) (Prasetyo & Pattiasina, 2015).

Tipe data MySQL, menurut Kustiyahningsih *(2011:147)* dalam (Firman et al., 2016), "Tipe data MySQL adalah data yang terdapat dalam sebuah tabel berupa *field – field* yang berisi nilai dari data tersebut. Nilai data dalam *field* memiliki tipe sendiri – sendiri".

MySQL dapat digunakan untuk membuat dan mengola *database* beserta isinya. Kita dapat memanfaatkan MySQL untuk menambahkan, mengubah dan menghapus data yang berada dalam *database* (Ratnasari, 2017).

2.5.2.1 Keistimewaan MySQL

MySQL memiliki beberapa keistimewaan, yaitu sebagai berikut.

1. *Portability*

   MySQL dapat berjalan stabil pada berbagai macam sistem operasi, antara lain *Windows, Linux, FreeBSD, Mac OS X Server, Solaris,* dan *Amig.*
2. *Open Source*

   Dibawah lisensi GPL, kita dapat menggunakan *software* SQL ini secara cuma-cuma atau gratisan tanpa dipungut biaya.
3. *Multiuser*

   MySQL dapat diakses *client* dalam waktu bersamaan tanpa menyebabkan konflik.
4. *Performent Tuning*

   SQL memiliki kecepatan yang tinggi dalam menangani setiap *query*.
5. *Colum Tuning*

MySQL memiliki tipe kolom yang sangat komplek, seperti *signed/unsigned, integer, float, double, char, vachar, text, blob*, dan lain-lain.

6. *Command and Function*

   MySQL memiliki *operator* dan fungsi secara penuh yang mengandung perintah *SELECT* dan *WHERE* dalam *query.*

7. *Security*

   MySQL memiliki beberapa lapisan keamanan seperti *level subnetmask*, izin akses *user* dengan perizinan yang mendetail, serta *password* yang terenkripsi.

8. *Scalability* dan *Limits*

   MySQL mampu menangani *database* dalam skala besar, dengan jumlah *record* lebih dari 50 juta, 60 ribu tabel, serta 5 miliar baris. Selain itu, batas *indeks* yang dapat ditampung mencapai 32 *indeks* pada setiap tabelnya.

9. *Connectivity:*

   Jika terletak pada jaringan maka MySQL dapat melakukan koneksi dengan *client* menggunakan *protocol TCP/IP*, *Unix socet (Unix*), atau *Namad Pipes (NT).*

10. *Localisation*

    MySQL dapat mendeteksi kesalahan (*error code*) pada *clien*t menggunakan lebih dari dua puluh bahasa. Meskipun demikian bahasa Indonesia belum termasuk didalamnya.

11. *Interface*

    MySQL memiliki *interface* (antarmuka) berbagai aplikasi dan bahasa pemograman menggunakan fungsi API (*Aplication Programing Interface).*

12. *Client* and *Tools*

    MySQL dilengkapi dengan berbagai *tool* yang dapat digunakan untuk administrasi *database*, dan pada tiap-tiap toolnya disertakan petunjuk *online.*

13. Struktur Tabel

    MySQL memiliki struktur *table* yang lebih *fleksibel* dalam menanganni *ALTER TABLE*, dibandingkan dengan *database* lainya seperti *PostgreSQL* dan *Oracle.*

    (S. Handayani & Kanedi, 2014)

2.5.2.2 Kelemahan MySQL

Di samping mempunyai keistimewaan, MySQL juga memiliki kelemahan, yaitu sebagai berikut.

1. Kurang mendukung koneksi bahasa pemrograman seperti *Visual Basic* atau biasa kita kenal dengan sebutan VB, *Foxpro*, *Delphi*, dan lain-lain sebab koneksi ini menyebabkan *field* yang dibaca harus sesuai dengan koneksi dari bahasa pemrograman *visual* tersebut.
2. Data yang dapat ditangani belum besar dan belum mendukung *windowing function*.

   (Utara, n.d.)

2.6 Aplikasi yang Digunakan

2.6.1 *Adobe Dreamweaver*

Menurut Firdaus (2007:15) dalam (Pada Sistem, Inventori, Pangan, & Sejahtera, 2017) mengemukakan bahwa "*Macromedia Dreamweaver 8* adalah sebuah perangkat lunak yang dapat digunakan oleh setiap orang untuk belajar bagaimana membuat *web* dengan mudah. Cara penggunaanya juga sangat sederhana dan gampang dimengerti".

Salah satu kekuatan *Macromedia Dreamweaver 8* ini adalah kemampuannya mendukung pemrograman *script server side* seperti *Active Server Pages* (ASP), ASP.NET, *ColdFusion*, *Java Server Pages* (JSP) dan PHP. Selain itu, tentunya mendukung pemrograman *client side* yang sangat terkenal dan banyak dipakai orang, yakni HTML dan *JavaScript* (Pada Sistem et al., 2017).

2.6.1.1 Fungsi *Adobe Dreamweaver*

Adobe Dreamweaver mempunyai berbagai macam kegunaan, sebagai berikut.

1. Untuk mendesain situs *web*.
2. Untuk membuat program berbasis *web*.
3. Untuk membuat *template blog*.

4. Untuk membuat situs *web* tanpa bersentuhan langsung dengan bahasa pemrograman.
   (Agustian, n.d.)

2.6.1.2 Kelebihan *Adobe Dreamweaver*

1. Dapat membuat kerangka *website* dengan mudah dan cepat.
2. Tersedia berbagai macam tema/*template*.
3. Memiliki 3 tampilan yaitu *Code View, Design View* dan *Split View*.
4. Memiliki Fitur *Preview/Live View*.
5. Kode yang dihasilkan ditulis secara rapi.
6. Memiliki alat alat khusus untuk membuat program berbasis *web*.
7. Mudah dioperasikan oleh pemula.
8. Memiliki banyak *plugin*.
   (Agustian, n.d.)

2.6.1.3 Kekurangan *Adobe Dreamweaver*

1. Harga *software* original-nya mahal.
   (Agustian, n.d.)

2.6.2 XAMPP

XAMPP adalah perangkat lunak bebas, yang mendukung banyak *system* operasi, merupakan kompilasi dari beberapa program. XAMPP merupakan *tool* yang menyediakan paket perangkat lunak ke dalam satu buah paket. Dengan meng*install* XAMPP maka tidak perlu lagi melakukan instalasi dan konfigurasi *web server Apache*, PHP dan MySQL secara manual. XAMPP akan menginstalasi dan mengkonfigurasikannya secara otomatis untuk anda atau *auto* konfigurasi (Ratnasari, 2017).

Bagian-bagian XAMPP, sebagai berikut.

1. X , kenapa disebut dengan *system* operasi? karena XAMPP bisa dijalankan di 4 OS besar yang sering digunakan oleh pengguna komputer saat ini. Dan 4

OS tersebut tidak lain dan tidak bukan adalah *Windows, Linux, Mac OS* dan *Solaris*.

2. A (*Apacahe*) merupakan aplikasi *web server*. *Apache* ini bersifat *open source* yang berarti gratis dan bisa di edit oleh penggunanya. Tugas utama *Apache* adalah menghasilkan halaman *web* yang benar kepada *user* berdasarkan kode PHP yang dituliskan oleh pembuat halaman *web*. Jika di perlukan juga berdasarkan kode PHP yang di tuliskan, maka dapat saja suatu *database* diakses terlebih dahulu (misalnya dalam MySQL) untuk mendukung halaman *web* yang di hasilkan.
3. M (MySQL), merupakan aplikasi *database server*. Perkembangannya disebut SQL yang merupakan kepanjangan dari *Structured Query Language*. SQL merupakan bahasa terstruktur yang digunakan untuk mengolah *database*. MySQL dapat digunakan untuk membuat dan mengelola *database* beserta isinya. Kita dapat memanfaatkan MySQL untuk menambahkan, mengubah, dan menghapus data yang berada dalam *database*.
4. P (PHP), bahasa pemrograman *web*. Bahasa pemrograman PHP merupakan bahasa pemrograman untuk membuat *web* yang bersifat *server-side scripting*. PHP memungkinkan kita untuk membuat halaman *web* yang bersifat dinamis. Sistem manajemen basis data yang sering digunakan bersama PHP adalah MySQl. Namun PHP juga mendukung sistem manajemen *database Oracle, Microsoft Access, Interbase, d-base, PostgreSQL*, dan sebagainya.
5. P (*Perl*), bahasa pemrograman, pertama kali dikembangkan oleh Larry Wall di mesin *Unix*. *Perl* pertama kali di rilis pada tanggal 18 Desember 1987 ditandai dengan keluarnya *Perl* 1. Dua di antara karakteristik utama *perl* adalah penanganan teks dan berbagai jalan pintas untuk meyelesaikan persoalan-persoalan umum. *Perl* sangat populer di gunakan dalam program-program CGI (*Common Gateway Interface*) dan protokol internet lainnya.
   (H. Handayani, n.d.)

2.7 Skripsi dan Jurnal Terkait

Berikut merupakan beberapa skripsi dan jurnal terdahulu tentang diagnosa penyakit *rubella*, dapat dilihat pada tabel 2.5 berikut ini.

Tabel 2.5 Skripsi dan Jurnal Terdahulu

No.	Judul	Penulis/Peneliti	Fakultas/Universitas	Tujuan/Identifikasi Masalah
1.	Tingkat Pengetahuan Ibu Tentang Imunisasi Tambahan Mr *(Measles Rubella)* pada Balita di Puskesmas Kotagede I Yogyakarta	1. Lailan Najah	Fakultas Ilmu Kesehatan/Universitas Aisyiyah Yogyakarta	Tujuan: Diketahuinya tingkat pengetahuan ibu tentang imunisasi tambahan MR pada balita di Puskesmas Kotagede I tahun 2017.
2.	Hubungan Pengetahuan Tentang Vaksin Mr *(Measles Rubella)* dan Pendidikan Ibu Terhadap Minat Keikutsertaan Vaksinasi Mr di Puskesmas Kartasura	1. Merlinta	Fakultas Kedokteran/ Universitas Muhammadiyah Surakarta	Tujuan: Penelitian ini bertujuan untuk mengetahui hubungan pengetahuan tentang vaksin MR dan pendidikan ibu terhadap minat keikutsertaan vaksinasi MR di puskesmas kartasura.

No.	Judul	Penulis/Peneliti	Fakultas/Universitas	Tujuan/Identifikasi Masalah
3.	Beberapa Faktor yang Berhubungan dengan Penerimaan Ibu Terhadap Imunisasi Measles Rubella pada Anak SD di Desa Gumpang, Kecamatan Kartasura, Kabupaten Sukoharjo	1. Gayuh Mustika Prabandari 2. Syamsulhuda Budi Musthofa 3. Aditya Kusumawati	Fakultas Kesehatan Masyarakat/Universitas Diponegoro	Identifikasi masalah: apa saja faktor yang berhubungan dengan penerimaan ibu terhadap imunisasi*Measles Rubella* di Desa Gumpang, Kecamatan Kartasura, Kabupaten Sukoharjo
4.	Analisis Penyebaran dan Genotipe Rubela di Jawa Barat Tahun 2011–2013	1. Acep T. Hardiana 2. Ardini S. Raksanaga 3. Rd. Tina D. Judistiani 4. Dyah Widhiastuti Novilia S. Bachtiar	Fakultas Kedokteran/Universitas Padjadjaran	Tujuan: Penelitian ini bertujuan untuk mengetahui penyebaran dan genotipe rubela di Jawa Barat dalam upaya pencegahan yang efektif.