BAB II

LANDASAN TEORI

2.1 Koperasi

Berdasarkan Peraturan Menteri Koperasi Dan Usaha Kecil Dan Menengah Republik Indonesia **NOMOR 16 /PER/M.KUKM/IX/2015** Tentang Pelaksanaan Kegiatan Usaha Simpan Pinjam Dan Pembiayaan Syariah Oleh Koperasi :

a. Koperasi adalah badan usaha yang beranggotakan orang-seorang atau badan hukum koperasi dengan melandaskan kegiatannya berdasarkan prinsip Koperasi sekaligus sebagai gerakan ekonomi rakyat yang berdasar atas asas kekeluargaan sebagaimana yang dimaksud dalam peraturan perundang-undangan perkoperasian.
b. Koperasi Simpan Pinjam dan Pembiayaan Syariah (KSPPS) adalah Koperasi yang kegiatan usahanya meliputi simpanan, pinjaman dan pembiayaan sesuai prinsip syariah, termasuk mengelola zakat, infaq/sedekah, dan wakaf.

2.2 Baitut Tamwil Muhammadiyah (BTM)

Secara etimologis *baitut tamwil* berarti rumah pembiayaan. Sehingga dikatakan *baitut tamwil* sebagai usaha pengumpulan dan penyaluran dana komersial. *Baitut Tamwil* adalah lembaga keuangan yang kegiatan utamanya menghimpun dana masyarakat dalam bentuk tabungan (simpanan) maupun deposito dan menyalurkannya kembali kepada masyarakat dalam bentuk

pembiayaan berdasarkan prinsip syariah melalui mekanisme yang lazim dalam dunia perbankan. (Huda & Heykal, 2013)

2.3 Kredit

Kredit berasal dari bahasa Yunani *Credere* yang berarti kepercayaan atau dalam bahasa latin *Creditum* yang berarti kepercayaan akan kebenaran (Kasmir, 2014).

Adapun unsur – unsur yang terkandung dalam pemberian suatu fasilitas kredit adalah sebagai berikut :

a. Kepercayaan

   Kepercayaan merupakan suatu keyakinan bagi si pemberi kredit bahwa kredit yang diberikan (baik berupa uang, barang, atau jasa) benar – benar diterima kembali di masa yang akan datang sesuai jangka waktu kredit.

b. Kesepakatan

   Kesepakatan ini dituangkan dalam suatu perjanjian dimana masing –masing pihak menandatangani hak dan kewajibannya masing – masing.

c. Jangka Waktu

   Setiap kredit yang diberikan memiliki jangka waktu tertentu, jangka waktu ini mencakup masa pengembalian kredit yang telah disepakati.

d. Resiko

   Akibat adanya tenggang waktu maka pengembalian kredit akan memungkinkan suatu resiko tidak tertagihnya atau macet suatu pemberian kredit.

e. Balas Jasa

Bagi bank balas jasa merupakan keuntungan atau pendapatan atas pemberian suatu kredit yang kita kenal dengan nama bunga.

2.4 Data Mining

Data mining adalah proses menemukan pola yang menarik dan pengetahuan dari data yang berjumlah besar (Kamber, 2012).

Secara garis besar, data mining dikelompokkan menjadi 2 kategori utama, yaitu :

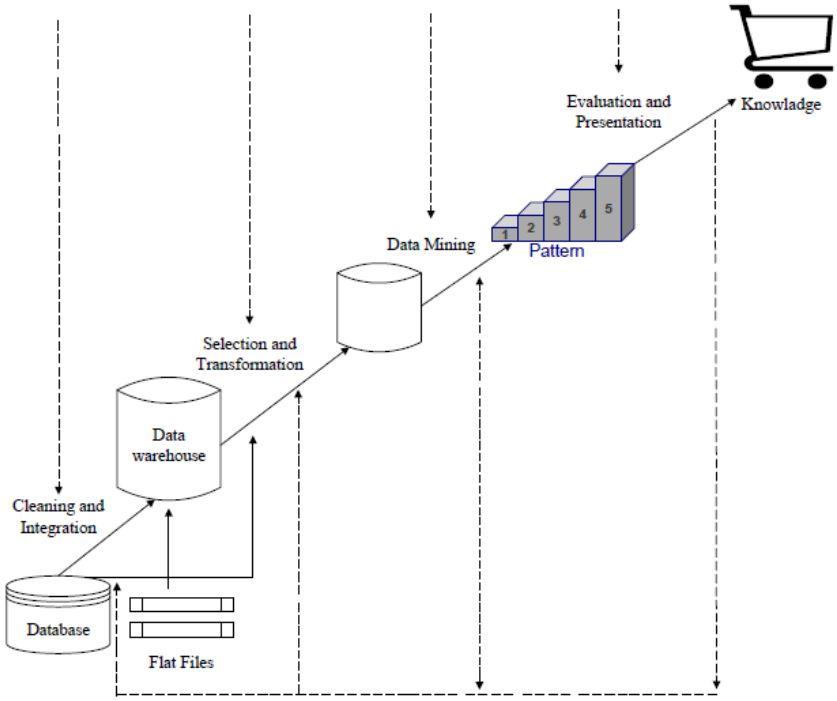
a. *Predictive* (Prediksi)

Predictive merupakan proses untuk menemukan pola dari data dengan menggunakan beberapa variabel lain di masa depan. Salah satu teknik yang terdapat dalam *predictive mining* adalah klasifikasi. Tujuan dari tugas prediktif adalah untuk memprediksi nilai dari atribut tertentu berdasarkan pada nilai atribut-atribut lain. Atribut yang diprediksi umumnya dikenal sebagai target atau variable tak bebas, sedangkan atribut-atribut yang digunakan untuk membuat prediksi dikenal sebagai *explanatory* atau *variable* bebas.

b. *Descriptive* (Deskripsi)

Descriptive dalam data mining merupakan proses untuk menemukan karakteristik penting dari data dalam suatu basis data. Tujuan dari tugas deskriptif adalah untuk menurunkan pola-pola (korelasi, *trend*, *cluster*, teritori, dan anomali) yang meringkas hubungan yang pokok dalam data. Tugas *data mining* deskriptif sering merupakan penyelidikan dan seringkali memerlukan teknik *post-processing* untuk validasi dan penjelasan hasil.

2.3.1 Tahap – tahap Data Mining



Gambar 1.1 Tahap – tahap Data Mining (Kamber, 2012)

Tahap – tahap tersebut, bersifat interaktif dimana pemakai terlibat langsung atau dengan perantaraan *knowledge base*.

1. Pembersihan data

   Pembersihan data merupakan proses menghilangkan noise dan data yang tidak konsisten atau data tidak relevan. Pada umumnya data yang diperoleh, baik dari *database* suatu perusahaan maupun hasil eksperimen, memiliki isian-isian yang tidak sempurna seperti data yang hilang, data yang tidak valid atau juga hanya sekedar salah ketik. Selain itu, ada juga atribut-atribut data yang tidak relevan dengan hipotesa *data mining* yang dimiliki. Data-data yang tidak relevan itu juga lebih baik dibuang. Pembersihan data juga akan mempengaruhi

performasi dari teknik *data mining* karena data yang ditangani akan berkurang jumlah dan kompleksitasnya.

2. Integrasi data

   Integrasi data merupakan penggabungan data dari berbagai *database* ke dalam satu *database* baru. Tidak jarang data yang diperlukan untuk *data mining* tidak hanya berasal dari satu *database* tetapi juga berasal dari beberapa *database* atau file teks. Integrasi data dilakukan pada atribut-aribut yang mengidentifikasikan entitasentitas yang unik seperti atribut nama, jenis produk, nomor pelanggan dan lainnya. Integrasi data perlu dilakukan secara cermat karena kesalahan pada integrasi data bisa menghasilkan hasil yang menyimpang dan bahkan menyesatkan pengambilan aksi nantinya. Sebagai contoh bila integrasi data berdasarkan jenis produk ternyata menggabungkan produk dari kategori yang berbeda maka akan didapatkan korelasi antar produk yang sebenarnya tidak ada.

3. Seleksi data

   Data yang ada pada *database* sering kali tidak semuanya dipakai, oleh karena itu hanya data yang sesuai untuk dianalisis yang akan diambil dari *database*. Sebagai contoh, sebuah kasus yang meneliti faktor kecenderungan orang membeli dalam kasus *market basket analysis,* tidak perlu mengambil nama pelanggan, cukup dengan id pelanggan saja.

4. Transformasi data

   Data diubah atau digabung ke dalam format yang sesuai untuk diproses dalam *data mining*. Beberapa metode *data mining* membutuhkan format data yang khusus sebelum bisa diaplikasikan. Sebagai contoh beberapa metode standar seperti analisis asosiasi dan *clustering* hanya bisa menerima input data kategorikal. Karenanya data berupa angka numerik yang berlanjut perlu dibagi-bagi menjadi beberapa interval. Proses ini sering disebut transformasi data.

5. Proses *mining*

   Merupakan suatu proses utama saat metode diterapkan untuk menemukan pengetahuan berharga dan tersembunyi dari data.

6. Presentasi pengetahuan

   Untuk mengidentifikasi pola-pola menarik kedalam *knowledge based* yang ditemukan. Dalam tahap ini hasil dari teknik *data mining* berupa pola-pola yang khas maupun model prediksi dievaluasi untuk menilai apakah hipotesa yang ada memang tercapai. Bila ternyata hasil yang diperoleh tidak sesuai hipotesa ada beberapa alternatif yang dapat diambil seperti menjadikannya umpan balik untuk memperbaiki proses *data mining*, mencoba metode *data mining* lain yang lebih sesuai, atau menerima hasil ini sebagai suatu hasil yang di luar dugaan yang mungkin bermanfaat.

2.5 Naïve Bayes

Naïve Bayes merupakan teknik prediksi berbasis probabilistic sederhanayang berdasar pada penerapan teorema Bayes (aturan Bayes) dengan

asumsi independensi (ketidaktergantungan) yang kuat (naif). Dengan kata lain, dalam Naïve Bayes model yang digunakan adalah "model fitur independen" (Prasetyo, 2012).

Pengklasifikasi Bayesian memiliki tingkat kesalahan minimal dibandingkan dengan klasifikasi lainnya. Namun, dalam prakteknya hal ini tidak selalu terjadi, karena ketidak akuratan asumsi yang dibuat untuk penggunanya, seperti kondisi kelas tidak ketergantungan, dan kurangnya data probabilitas yang tersedia. Pengklasifikasi Bayesian juga berguna dalam memberikan pembenaran teoritis untuk pengklasifikasi lain yang tidak secara eksplisit menggunakan teorema bayes.

Menurut **Han dan Kamber (2011:351)** Proses dari Bayesian *The Naïve Bayes Classifier* atau *Simple Bayesian Classifier* sebagai berikut :

1. *Variable* D menjadi pelatihan set *tuple* dan *label* yang terkait dengan kelas. Seperti biasa, setiap *tuple* diwakili oleh vektor atribut ndimensi, X = (x1, x2, ..., xn), ini menggambarkan pengukuran n dibuat pada *tuple* dari atribut n, masing-masing, A1, A2, ..., An.
2. Misalkan ada kelas m, C1, C2, ..., Cm. Diberi sebuah *tuple*, X, *classifier* akan memprediksi X yang masuk kelompok memiliki probabilitas *posterior* tertinggi, kondisi-disebutkan pada X. Artinya, *classifier naive bayesian* memprediksi bahwa X *tuple* milik kelas Ci jika dan hanya jika :

$$P(C_i|\boldsymbol{X}) > P(C_j|\boldsymbol{X}) \quad \text{for } 1 \leq j \leq m, j \neq i.$$

Jadi memaksimalkan P (Ci | X). Ci kelas yang P (Ci | X) dimaksimalkan disebut hipotesis posteriori maksimal.

Dengan teorema Bayes:

$$P(C_i|\boldsymbol{X}) = \frac{P(\boldsymbol{X}|C_i)P(C_i)}{P(\boldsymbol{X})}.$$

Keterangan :

P(Ci|X) = Probabilitas hipotesis Ci jika diberikan fakta atau *record* X *(Posterior probability)*

P(X|Ci) = mencari nilai parameter yang memberi kemungkinan yang paling besar *(likehood)*

P(Ci) = *Prior probability* dari X *(Prior probability)*

P(X) = Jumlah *probability tuple* yg muncul

3. Ketika P (X) adalah konstan untuk semua kelas, hanya P (X | Ci) P(Ci) butuh dimaksimalkan. Jika probabilitas kelas sebelumnya tidak diketahui, maka umumnya diasumsikan ke dalam kelas yang sama, yaitu, P (C1) = P (C2) = · · · = P (Cm), maka dari itu akan memaksimalkan P (X | Ci). Jika tidak, maka akan memaksimalkan P (X | Ci) P (Ci). Perhatikan bahwa probabilitas sebelum kelas dapat diperkirakan oleh P (Ci) = | Ci, D | / | D |, dimana | Ci, D | adalah jumlah *tuple* pelatihan kelas Ci di D.
4. Mengingat *dataset* mempunyai banyak atribut, maka akan sangat sulit dalam mengkomputasi untuk menghitung P(X|Ci). Agar dapat mengurangi perhitungan dalam mengevaluasi P(X|Ci), asumsi *naïve* independensi kelas bersyarat dibuat. Dianggap bahwa nilai-nilai dari atribut adalah kondisional independen satu sama lain, diberikan kelas *label* dari *tuple* (yaitu bahwa tidak ada hubungan ketergantungan diantara atribut) dengan demikian :

$$P(\boldsymbol{X}|C_i) = \prod_{k=1}^{n} P(x_k|C_i)$$
$$= P(x_1|C_i) \times P(x_2|C_i) \times \cdots \times P(x_n|C_i).$$

Maka dapat dengan mudah memperkirakan probabilitas P (x1 | Ci), P (x2 | Ci),. . . , P (xn | Ci) dari pelatihan *tuple.* Ingat bahwa di sini xk mengacu pada nilai atribut Ak untuk *tuple* X. Untuk setiap atribut, dilihat dari apakah atribut tersebut kategorikal atau *continuous-valued* . Misalnya, untuk menghitung P (X | Ci) mempertimbangkan hal-hal berikut:

a. Jika Ak adalah kategorikal, maka P (Xk | Ci) adalah jumlah *tuple* kelas Ci di D memiliki nilai Xk untuk atribut Ak, dibagi dengan | Ci, D |, jumlah *tuple* kelas Ci di D.
b. Jika Ak *continuous-valued* , maka perlu melakukan sedikit lebih banyak pekerjaan, tapi perhitunganya cukup sederhana. Sebuah atribut *continuous-valued* biasanya diasumsikan memiliki distribusi *Gaussian* dengan rata-rata μ dan standar deviasi σ, didefinisikan oleh:

$$g(x, \mu, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}},$$

Sehingga :

$$P(x_k|C_i) = g(x_k, \mu_{C_i}, \sigma_{C_i}).$$

Setelah itu hitung μCi dan σCi, yang merupakan deviasi *mean* (rata-rata) dan standar masing-masing nilai atribut k untuk *tuple* pelatihan kelas Ci. Setelah itu gunakan kedua kuantitas dalam Persamaan, bersama-sama dengan xk, untuk memperkirakan P (xk | Ci).

5. Untuk memprediksi *label* kelas x, P(X|Ci)P(Ci) dievaluasi untuk setiap kelas Ci. *Classifier* memprediksi kelas *label* dari *tuple* x adalah kelas Ci, jika :

$$P(\boldsymbol{X}|C_i)P(C_i) > P(\boldsymbol{X}|C_j)P(C_j) \quad \text{for } 1 \leq j \leq m, j \neq i.$$

Dengan kata lain, label kelas diprediksi adalah Ci yang mana P (X | Ci) P (Ci) adalah maksimal.

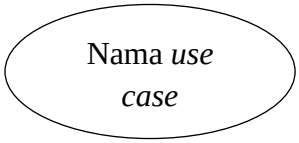
2.6 UML (*Unified Modelling Language*)

UML (*Unified Modelling Language*) adalah salah satu standar bahasa yang banyak digunakan didunia industry untuk mendefinisikan *requirement*, membuat analisis & desain, serta menggambarkan arsitektur dalam pemrograman berorientasi objek (Sukamto & Shalahuddin, 2013).

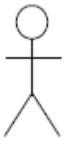
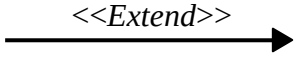
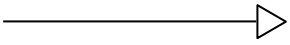
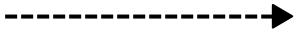
2.6.1 *Use Case Diagram*

Use case atau diagram *use case* merupakan pemodelan untuk kelakuan (*behavior*) sistem informasi yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat.

Tabel 2.1 Simbol – Simbol Diagram *Use Case*

No.	Simbol	Keterangan
1.	*Use Case* 	Fungsionalitas yang disediakan sistem sebagai unit – unit yang saling bertukar pesan antar unit atau aktor; biasanya dinyatakan dengan menggunakan kata kerja di awal frase nama *use case.*

Tabel 2.1 Simbol – Simbol Diagram *Use Case*

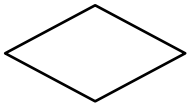
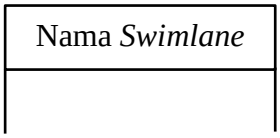
No.	Simbol	Keterangan
2.	Aktor / *Actor*  nama aktor	Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat di luar sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang; biasanya dinyatakan menggunakan kata benda di awal frase nama *actor.*
3.	Asosiasi / *association* ———————	Komunikasi antar aktor dan *use case* yang berpartisipasi pada *use case* atau *use case* memiliki interaksi dengan actor.
4.	Ekstensi / *Extend* <<*Extend*>> 	Relasi *use case* tambahan ke sebuah *use case* dimana *use case* yang ditambahkan dapat berdiri sendiri walau tanpa *use case* tambahan itu; biasanya *use case* tambahan memiliki nama depan yang sama dengan *use case* yang ditambahkan.
5.	Generalisasi/*Generalization* 	Hubungan generalisasi dan spesialisasi (umumkhusus) antara dua buah *use case* dimana fungsi yang satu adalah fungsi yang lebih umum dari lainnya.
6.	Menggunakan *Include* / *Uses* <<*include*>> 	Relasi *use case* tambahan ke sebuah *use case* di mana *use case* yang ditambahkan memerlukan *use case* ini untuk menjalankan fungsinya atau sebagai syarat dijalankan *use case*. *Include* berarti *use case* yang ditambahkan akan selalu dipanggil saat *use case* tambahan dijalankan.

2.6.2 *Activity Diagram*

Diagram Aktivitas atau *activity diagram* menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis atau menu yang ada pada perangkat lunak. Diagram aktivitas

menggambarkan aktivitas sistem bukan apa yang dilakukan aktor, jadi aktivitas yang dapat dilakukan oleh sistem.

Tabel 2.2 Simbol – simbol diagram aktivitas

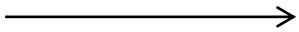
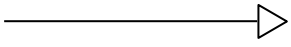
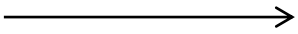
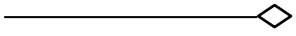
No.	Simbol	Keterangan
1.	Status Awal	Status awal aktivitas sistem, sebuah diagram aktivitas memiliki sebuah status awal.
2.	Aktivitas	Aktivitas yang dilakukan sistem, aktivitas biasanya diawali dengan kata kerja.
3.	Percabangan / *Decision* 	Asosiasi percabangan dimana jika ada pilihan aktivitas lebih dari satu.
4.	Penggabungan / *Join*	Asosiasi penggabungan dimana lebih dari satu aktivitas digabungkan menjadi satu.
5.	Status akhir	Status akhir yang dilakukan sistem, sebuah diagram aktivitas memiliki sebuah status akhir.
6.	*Swimlane* 	Memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi.

2.6.3 *Class Diagram*

Diagram kelas atau *class diagram* menggambarkan struktur sistem dari segi pendefinisian kelas – kelas yang akan dibuat untuk membangun sistem. Kelas memiliki apa yang disebut atribut dan metode atau operasi.

1. Atribut merupakan variabel – variabel yang dimiliki oleh suatu kelas
2. Operasi atau metode adalah fungsi – fungsi yang dimiliki oleh suatu kelas

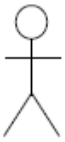
Tabel 2.3 Simbol – Simbol Diagram Kelas

No.	Simbol	Keterangan
1.	Kelas nama_kelas + attribut + operasi()	Kelas pada struktur sistem.
2.	Antarmuka / *interface* nama_interface	Sama dengan konsep *interface* dalam pemrograman berorientasi objek.
3.	Asosiasi / *association* 	Relasi antarkelas dengan makna umum, asosiasi biasanya juga disertai dengan *multiplicity.*
4.	Asosiasi berarah / *directed association* 	Relasi antar kelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi biasanya juga disertai *multiplicity.*
5.	Generalisasi 	Relasi antarkelas dengan makna generalisasi - spesialisasi (umum khusus)
6.	Kebergantungan/*dependency* 	Relasi antarkelas dengan makna kebergantungan antarkelas
7.	Agregasi / *aggregation* 	Relasi antarkelas dengan makna semua bagian (*whole-part*)

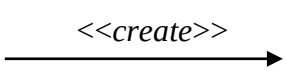
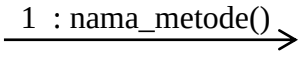
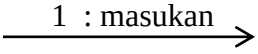
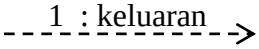
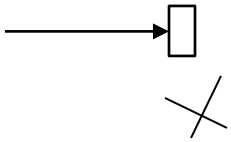
2.6.4 *Sequence Diagram*

Sequence diagram menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan *message* yang dikirimkan dan diterima antar objek. Banyaknya diagram sekuen yang harus digambar adalah minimal sebanyak pendefinisian *use case* yang memiliki proses sendiri atau yang penting semua *use case* yang telah didefinisikan interaksi jalannya pesan sudah dicakup pada diagram sekuen sehingga semakin banyak *use case* yang didefinisikan maka diagram sekuen yang harus dibuat juga semakin banyak.

Tabel 2.4 Simbol – simbol *sequence diagram*

No.	Simbol	Keterangan
1.	Aktor / *Actor*  nama aktor atau Nama aktor tanpa waktu aktif	Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat di luar sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang; biasanya dinyatakan menggunakan kata benda di awal frase nama *actor*.
2.	Garis hidup / *lifeline* 	Menyatakan kehidupan suatu objek
3.	Objek Nama objek : nama kelas	Menyatakan objek yang berinteraksi pesan

Tabel 2.4 Simbol – simbol *sequence diagram*

No.	Simbol	Keterangan
4.	Waktu aktif	Menyatakan objek dalam keadaan aktif dan berinteraksi, semua yang terhubung dengan waktu aktif ini adalah sebuah tahapan yang dilakukan didalamnya.
5.	Pesan tipe *create* 	Menyatakan suatu objek membuat objek yang lain, arah panah mengarah pada objek yang dibuat.
6.	Pesan tipe *call* 	Menyatakan suatu objek memanggil operasi/metode yang ada pada objek lain atau dirinya sendiri.
7.	Pesan tipe *send* 	Menyatakan bahwa suatu objek mengirimkan data/ masukan/ informasi ke objek lainnya, arah panah mengarah pada objek yang dikirimi
8.	Pesan tipe *return* 	Menyatakan bahwa suatu objek yang telah menjalankan suatu operasi atau metode menghasilkan suatu kembalian ke objek tertentu, arah panah mengarah pada objek yang menerima kembalian.
9.	Pesan tipe *destroy* 	Menyatakan suatu objek mengakhiri hidup objek yang lain, arah panah mengarah pada objek yang diakhiri, sebaiknya jika ada *create* maka ada *destroy.*

2.7 PHP

PHP (*HyperText Preprocessor*) merupakan bahasa utama *script server side* yang disisipkan pada HTML yang dijalankan di *server*, dan juga bisa digunakan untuk membuat aplikasi dekstop (Sidik, 2014).

Kelebihan-kelebihan dari PHP diantaranya adalah:

a. PHP mudah dibuat dan dijalankan, maksudnya PHP dapat berjalan dalam Web Server dan dalam Sistem Operasi yang berbeda pula.
b. PHP adalah *software open-source* yang gratis dan bebas didistribusikan kembali di bawah lisensi GPL (*GNU Public License). User* dapat men*download* kode-kode PHP tanpa harus mengeluarkan uang atau khawatir dituntut oleh pihak pencipta PHP.
c. PHP bisa dioperasikan pada *platform* Linux ataupun Windows.
d. PHP sangat efisien, karena PHP hanya memerlukan *resource system* yang sedikit disbanding dengan bahasa pemrograman lain.
e. Ada banyak *Web Server* yang mendukung PHP, seperti Apache, PWS, IIS, dan lain-lain.
f. PHP juga didukung oleh banyak database, seperti MySQL, PostgreSQL, Interbase, SQL, dan lain-lain.
g. Bahasa pemrograman PHP sintaknya sederhana, singkat dan mudah untuk dipahami.
h. HTML-*embedded,* artinya PHP adalah bahasa yang dapat ditulis dengan menempelkan pada sintak-sintak HTML.

2.8 MySQL

Database berfungsi sebagai penampung data yang diinputkan melalui form *website*. Selain itu juga dapat dibalik dengan menampilkan data yang tersimpan dalam database ke dalam halaman *website.* Jenis database yang sangat popular dan digunakan pada banyak *website* di internet sebagai bank data adalah MySQL. MySQL menggunakan SQL yang bersifat gratis, selain itu MySQL dapat berjalan di berbagai platform, antara lain Linux, Windows,

dll. Database MySQL dapat dibuat dan diolah menggunakan tampilan jendela *phpmyadmin* atau menggunakan *script* pemrograman PHP (Madcom, 2013).

Beberapa keunggulan dari MySQL yaitu:

a. Cepat, handal dan Mudah dalam penggunaannya

   MySQL lebih cepat tiga kali atau empat kali dari pada database *server* komersial yang beredar saat ini, mudah diatur dan tidak memerlukan seseorang ahli untuk mengatur administrasi pemasangan MySQL.

b. Didukung oleh berbagai bahasa

   Database *server* MySQL dapat memberikan pesan *error* dalam berbagai bahasa seperti Belanda, Portugis, Spanyol, Inggris, Perancis, Jerman, dan Italia.

c. Mampu membuat tabel berukuran sangat besar

   Ukuran maksimal dari setiap tabel yang dapat dibuat dengan MySQL adalah 4 GB sampai dengan ukuran file yang dapat ditangani oleh sistem operasi yang dipakai.

d. Lebih murah

   MySQL bersifat *open source* dan didistribusikan dengan gratis tanpa biaya untuk UNIX *platform*, OS/2 dan *Windows platform.*

e. Melekatnya integrasi PHP dengan MySQL

   Keterikatan antara PHP dengan MySQL yang sama-sama *software opensource* sangat kuat, sehingga koneksi yang terjadi lebih cepat jika dibandingkan dengan menggunakan database *server* lainnya. Modul

MySQL di PHP telah dibuat *built-in* sehingga tidak memerlukan konfigurasi tambahan pada *file* konfigurasi PHP ini.