BAB II

LANDASAN TEORI

2.1. Ruang Lingkup Pengajaran Berbantuan Komputer

Pengajaran berbantuan komputer telah berkembang cukup lama, dari bentuk yang sederhana, seperti *drill and practice* dan tutorial sampai ke bentuk yang kompleks seperti pengajaran berbantuan komputer yang bersifat cerdas. Untuk mendapatkan hasil yang optimal, perangkat ajar di dalam pengajaran berbantuan komputer harus memberikan kemungkinan-kemungkinan sehingga pemakainya dapat mengolah informasi sedalam mungkin. Pengolahan informasi yang dalam akan sangat membantu penyimpanan informasi di dalam memori seseorang, yang akan mempengaruhi jangkan waktu ingatan akan memori tersebut. Pengolahan informasi ini sangat dipengaruhi oleh rancangan instruksional dari perangkat ajar dan pemakaian strategi belajar yang tepat oleh pemakai perangkat ajar.

"*Pengertian PBK ialah sistem komputer yang dipergunakan untuk menyampaikan pengajaran kepada siswa (user), melalui interaksi secara langsung antara siswa (user) dengan materi pelajaran yang di program ke dalam sistem komputer. Penggunaan metode PBK diharapkan dapat mendukung fungsi buku dalam menyampaikan suatu materi kepada pihak yang mempelajari materi tersebut*." (Anis Gunaefi, 1997)

Ada beberapa jenis umum Pengajaran Berbantuan Komputer (PBK), yaitu:

1. Tutorial

   Tutorial adalah program komputer yang mengajar dengan berusaha berdialog dengan siswa. Disajikan berupa materi-materi. Dapat dikatakan bahwa tutorial adalah PBK yang paling mendasar.

2. Latihan (*Drill*)

   Dalam latihan, sekumpulan pertanyaan atau masalah disajikan berulang-ulang sampai siswa mampu menjawab dan menyelesaikan masalah yang diberikan pada suatu tingkat yang sudah ditentukan sebelumnya. Program komputer ini diharapkan mampu meningkatkan keefektifan, efisiensi dan kenyamanan suatu *drill*.

3. Simulasi

   Simulasi pengajaran adalah program komputer untuk mengajar siswa yang menggambarkan suatu fenomena yang dipelajari. Simulasi adalah pendekatan yang mendekati nilai kreativitas tinggi dan pendekatan yang menarik untuk diterapkan pada PBK. Program demonstrasi (visualisasi) adalah bagian dari simulasi, dengan fungsi hanya memperlihatkan sesuatu pada siswa.

4. Permainan

   Diantara metode pendidikan yang penting adalah permainan. Permainan merupakan hal yang menyenangkan dan dilakukan oleh setiap orang dan kualitas motivasinya dapat meningkatkan keinginan belajar.

5. Test

Evaluasi merupakan komponen penting dalam proses pengajaran. Program komputer untuk program pengevaluasian bukan hanya memungkinkan kualitas evaluasi, yaitu meningkatkan keakuratan dalam pengukuran pengetahuan siswa. Tetapi memungkinkan pula membebaskan pengajar dalam administrasi dan penilaian evaluasi, yang kadang membosankan.

6. PBK Cerdas

PBK Cerdas memiliki dua area utama yaitu model siswa dan sistem pakar. Program model siswa merangkum model data rinci tentang kerja siswa dan memformulasikan suatu hipotesa tentang apa yang siswa tahu dan pikir. Dari informasi tersebut memutuskan strategi pengajaran dan urutannya.

Untuk sistem pakar, idealnya memiliki semua pengetahuan tentang hal tertentu, himpunan logika yang menghubungkan semua informasi sehingga bukan hanya daftar fakta tetapi pengetahuan yang terintegrasi. Dalam beberapa kasus sistem pakar adalah suatu program yang membolehkan seseorang untuk berkomunikasi dengan mudah dan menggunakan bahasa alami dengan komputer.

2.2. Pengertian Database

Sebuah sistem *database* meliputi beberapa komponen yang secara bersama-sama memberikan perbedaan tertentu dan karakteristik yang khusus. Berikut ini beberapa definisi ringkas dari *database* :

"Database adalah sistem file komputer yang menggunakan cara perorganisasian file tertentu, dimaksudkan untuk mempercepat pembaruan masing-masing record, serta pembaruan secara serempak atas record terkait, juga untuk mempermudah dan mempercepat akses terhadap seluruh record lewat program-program aplikasi, serta akses yang cepat terhadap data yang tersimpan yang harus digunakan secara bersama-sama untuk dibaca guna penyusunan laporan-laporan rutin atau khusus ataupun untuk penyelidikan." (Azhar Susanto, 2000).

"Database merupakan kumpulan data-data yang tersimpan di dalam media penyimpanan disuatu perusahaan (arti luas) atau di dalam komputer (arti sempit)." (Andri Kristianto, 2004).

Dari pengertian diatas maka dapat disimpulkan bahwa *database* memiliki beberapa kriteria penting, yaitu :

1. Bersifat data *oriented* dan bukan program *oriented*.
2. Dapat digunakan oleh beberapa program aplikasi tanpa mengubah basis datanya.
3. Dapat berkembang dengan mudah, baik itu volume maupun strukturnya.
4. Dapat memenuhi kebutuhan sistem-sistem baru secara mudah.
5. Dapat digunakan dengan cara yang berbeda.
6. Kerangkapan data (*Data Redudancy*) minimal.

2.2.1. Perancangan Database

Dalam perancangan *database* terbagi dalam beberapa tahap sebagai berikut :

1. Pemodelan data, proses pembangunan *database* untuk menentukan tabel mana yang dibutuhkan suatu program aplikasi.
2. Pemantapan sasaran ke dalam proses operasional, untuk meyakinkan sasaran dari perancangan *database* diperlukan pemantapan yang membuat proses operasional lebih terperinci kedalam suatu *database* sederhana.
3. Penentuan objek *database* yang dibutuhkan.
4. Memeriksa keberadaan *database* sesuai dengan tipenya, pemeriksaan ini berguna pada saat pembuatan normalisasi *database* pada suatu sistem.
5. Membuat bingkai kerja (*framework*) *database*, pada tahap ini pembuatan bingkai *database* terbagi kedalam beberapa bagian yaitu membuat nama *field* dengan tipenya dan pengefisienan ukuran *field*.
6. Pengorganisasian tabel dengan tipenya, proses ini membagi-bagi peletakan suatu *field* dalam *database*.
7. Penggambaran tabel berelasi, agar lebih mudah dalam perancangan *database* diperlukan suatu tabel sehingga dalam menentukan mana master data atau data tambahan.
8. Normalisasi *database*, normalisasi data merupakan eleminasi dari *database* yang berlebihan (penggunaan data lebih dari satu tabel) dari tiap tabel dalam suatu *database*.

2.3. Rekayasa Perangkat Lunak

Definisi menurut Fritz Bauer *"Rekayasa Perangkat Lunak adalah pengembangan dan penggunaan prinsip pengembangan suara untuk memperoleh perangkat lunak secara ekonomis yang reliable dan bekerja secara efisien pada mesin nyata"*.

Menurut IEEE [IEE93] mengembangkan definisi yang lebih komprehensif rekayasa, rekayasa perangkat lunak yaitu :

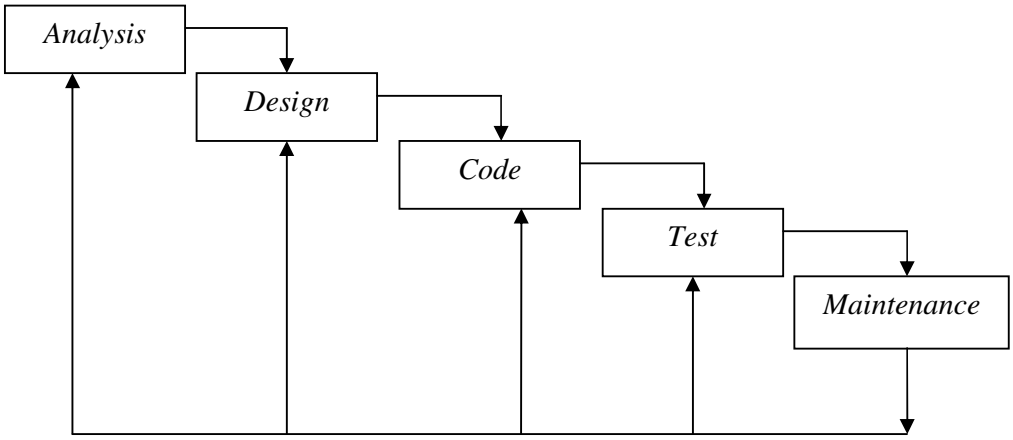
1. Aplikasi dari sebuah pendekatan *kuantifiabel*, disiplin, dan sistematis kepada pengembangan, operasi, dan pemeliharaan perangkat lunak.
2. Studi tentang pendekatan-pendekatan seperti pada nomor satu.

2.4. Rekayasa Sistem Berorientasi Objek

Rekayasa adalah bentuk upaya manusia untuk mencari atau merancang produk yang berkualitas. Produk dapat berupa sistem atau perangkat lunak yang sedang atau akan dibangun dengan suatu model pengembangan.

Orientasi objek adalah cara pandang bukan sekedar algoritma yang diimplementasikan pada bahasa pemrograman berorientasi objek, cara berpikir orientasi objek adalah segala sesuatu dipandang sebagai objek. Objek dapat berupa konsep, abstraksi atau sesuatu dengan batas-batas tegas dan mempunyai arti untuk persoalan yang ditangani. Objek mempunyai identitas dan dapat dibedakan. Ada dua kegunaan objek, yaitu untuk meningkatkan pemahaman dunia nyata dan menyediakan dasar pengetahuan praktis untuk implementasi komputer.

Model yang digunakan dalam proses perancangan perangkat lunak masih seperti model yang digunakan dalam metodologi konvensional, yaitu dapat menggunakan model *life cycle* atau *waterfall*. Dalam model *life cycle*, terdapat lima tahapan proses, yaitu tahap rekayasa sistem, analisis, perancangan, pemrograman, pengujian, pemeliharaan. Dengan menggunakan *waterfall*, memungkinkan adanya pengembangan aplikasi di kemudian hari, sehingga *software* yang dibangun dapat bertahan dalam jangka waktu yang panjang.



Gambar 2.1 Tahapan model waterfall

1. Rekayasa sistem

   Merupakan kegiatan untuk mempelajari dan menganalisis kebutuhan sistem, data yang mengalir dan perangkat penunjang seperti *software* dan *hardware* yang digunakan untuk membangun sistem.

2. Analisis

   Menentukan apakah kegiatan dari rekayasa sistem dapat diimplementasikan menjadi sebuah sistem atau tidak dan menentukan

prosedur-prosedur yang bekerja. Adapun prosedur - prosedur tersebut meliputi masukan (*input*), pemrosesan (*processing*), dan keluaran (*output*).

3. Perancangan

Tahap desain objek menggunakan desain berorientasi objek, secara spesifik mempunyai arti bahwa kita mengorganisasi perangkat lunak sebagai kumpulan dari objek tertentu yang memiliki struktur data dan perilakunya. Konsep berorientasi objek dapat dilakukan untuk semua sistem bahasa yang digunakan, yaitu UML (*Unified Modelling Language*), dan menggunakan beberapa diagram dari UML tersebut seperti :

a. *Use Case Diagram* : Diagram ini menjelaskan manfaat sistem jika dilihat menurut pandangan orang yang berada diluar sistem (*actor*) dan menunjukan fungsionalitas suatu sistem atau kelas dan bagaimanan sistem berinteraksi dengan dunia luar.

b. *Activity diagram* : Diagram ini memodelkan alur kerja (*workflow*) sebuah proses bisnis dan urutan aktifitas dalam suatu proses.

c. *Sequence diagram* : Diagram ini menunjukkan kolaborasi antara objek dan menunjukan bagaimana urutan operasi berlangsung, pesan apa yang dikirim, dan kapan pesan dikirimkan.

d. *Class Diagram* : Diagram ini menggambarkan struktur kelas-kelas dari suatu sistem dan memperlihatkan hubungan antar

kelas dan penjelasan detail tiap-tiap kelas didalam model desain dari suatu sistem.

4. Pengkodean

   Merupakan kegiatan yang mengimplementasikan hasil dari perancangan perangkat lunak ke dalam kode program agar pengguna (*user*) dapat memahami sistem yang sedang dibangun.

5. Pengujian

   Memfokuskan pada fungsi internal, fungsi eksternal dari perangkat lunak dan mencari segala kemungkinan kesalahan, memeriksa apakah input data sesuai dengan hasil yang diinginkan setelah proses.

6. Pemeliharaan

   Merupakan kegiatan untuk memelihara (*maintenance*) perangkat lunak yang telah dibangun, pemeliharaan tersebut dilakukan agar keutuhan program dapat terjaga.

2.5. Definisi Aliran Data Berorientasi Objek

Pemrograman berorientasi objek, atau sering juga disebut dengan OOP (*Object Oriented Programming*), merupakan paradigma baru dalam pengembangan sistem/perangkat lunak. Tidak seperti pendahulunya, yaitu pemrograman dengan teknik terstruktur (*structured programming*) yang sering mengalami kegagalan dalam hal sistem/perangkat lunak yang tidak sesuai dengan kebutuhan dan harapan pengguna (*user's need and expectations*) serta kerap kali

tidak dapat memenuhi jadwal yang telah ditentukan sebelumnya. Pemrograman berorientasi objek dengan paradigma barunya, menjanjikan tingkat penggunaan ulang (*reusability*), kinerja (*performance*), serta kehandalan (*reliability*) yang lebih tinggi. (Roger Pressman, 2000).

Saat ini, dunia pemrograman berorientasi objek mengenal UML (*Unified Modelling Language*) sebagai alat utama dalam analisis dan perancangan sistem. UML merupakan alat yang sangat sesuai dengan pemrograman berorientasi objek sebab konsep dasarnya adalah memodelkan kelas-kelas (beserta atribut serta operasi di dalamnya) bersamaan dengan relasi-relasi yang terjadi antar kelas yang bersangkutan.

2.6. Definisi UML (*Unified Modeling Language*)

UML (*Unified Modelling Language*) adalah sebuah "bahasa" yang telah menjadi standar dalam industri untuk visualisasi, merancang dan mendokumentasikan sistem piranti lunak. UML menawarkan sebuah standar untuk merancang model sebuah sistem.

UML adalah bahasa grafis untuk mendokumentasikan, menspesifikasikan dan membangun sistem perangkat lunak. UML berorientasi objek, menerapkan banyak level abstraksi, tidak bergantung proses pengembangan, tidak bergantung bahasa dan teknologi, pemaduan beberapa notasi di beragam metodologi, usaha bersama dari banyak pihak, didukung oleh kakas-kakas yang diintegrasikan oleh XML. Standar UML dikelola oleh OMG (*Object Management Group*).

UML (*Unified Modelling Language*) adalah bahasa pemodelan untuk menspesifikasikan, memvisualisasikan, membangun dan mendokumentasikan artifak-artifak dari sistem.

1. Di dalam *system intensive process*, metode diterapkan sebagai proses untuk menurunkan atau mengevolusikan sistem.
2. Sebagai bahasa, UML digunakan untuk komunikasi yaitu alat untuk menangkap pengetahuan mengenai satu subyek dan mengekspresikan pengetahuan (sintaks) yang memperdulikan subyek untuk maksud komunikasi. Subyek adalah sistem yang dibahas.
3. Sebagai bahasa pemodelan, UML fokus pada pemahaman subyek melalui formulasi model dari subyek. Model memuat pengetahuan pada subyek, dan aplikasi dari pengetahuan ini berkaitan dengan intelejensia.
4. Berkaitan dengan unifikasi.UML memadukan praktek rekayasa terbaik sistem informasi dan industri, meliputi beragam tipe sistem (perangkat lunak dan non perangkat lunak), domain (bisnis, perangkat lunak) dan proses siklus hidup.
5. begitu diterapkan untuk menspesifikasikan sistem, UML dapat digunakan untuk mengkomunikasikan “apa” yang diperlukan dari sistem dan “bagaimana” sistem dapat direalisasikan.
6. Begitu diterapkan untuk menvisualisasikan sistem, UML dapat digunakan untuk menjelaskan sistem secara visual sebelum direalisasikan.
7. Begitu diterapkan untuk membangun sistem, UML dapat digunakan untuk memandu realisasi sistem serupa dengan “blueprint”.

8. Begitu diterapkan untuk mendokumentasikan sistem, UML dapat digunakan untuk menangkap pengetahuan mengenai sistem pada seluruh siklus hidup.

UML bukanlah :

1. Bahasa pemrograman visual, tapi bahasa pemodelan visual.
2. Spesifikasi kakas, tapi spesifikasi bahasa pemodelan.
3. Proses, tapi yang memungkinkan proses-proses.

Tujuan utama perancangan UML adalah :

1. Menyediakan bahasa pemodelan visual yang ekspresif dan siap pakai untuk mengembangkan dan pertukaran model-model yang berarti.
2. Menyediakan mekanisme perluasan dan spesifikasi untuk memperluas konsep-konsep inti.
3. Mendukung spesifikasi independen bahasa pemrograman dan proses pengembangan tertentu.
4. Menyediakan basis formal untuk pemahaman bahasa pemodelan.
5. Mendorong pertumbuhan pasar kakas berorientasi objek.
6. Mendukung konsep-konsep pengembangan level lebih tinggi seperti komponen, kolaborasi, *framework* dan *pattern.*

2.6.1. Diagram Teknik Pemodelan UML

Diagram mengemukakan banyak hal, penggunaan notasi yang terdefinisi baik dan ekspresif adalah penting pada proses pengembangan perangkat lunak, yaitu :

1. Notasi standar memungkinkan pengembang mendeskripsikan skenario atau rumusan arsitektur dan kemudian mengkomunikasikan secara tidak ambigu.
2. Notasi yang bagus membebaskan otak untuk berkonsentrasi pada masalah-masalah yang lebih lanjut.
3. Notasi yang baik memungkinkan mengeleminasi keperluan pemeriksaan konsistensi dan kebenaran keputusan keputusan dengan menggunakan *tool* terotomatisasi.

2.6.2. Diagram Struktur

Diagram ini untuk memvisualisasi, menspesifikasikan, membangun dan mendokumentasikan aspek statik dari sistem. Diagram struktur di UML terdiri dari :

1. Diagram kelas (*Class diagram*)

   Diagram ini menunjukkan sekumpulan kelas, *interface* dan kolaborasi dan keterhubungannya.
2. Diagram objek (*Object diagram*)

   Diagram ini menunjukkan sekumpulan objek dan keterhubungannya.

3. Diagram komponen (*Component diagram*)

   Diagram ini menunjukkan organisasi dan kebergantungan di antara sekumpulan komponen. Diagram ini merupakan pandangan statik terhadap implementasi sistem.

4. Diagram deployment (*Deployment diagram*)

   Diagram ini menunjukkan konfigurasi pemrosesan saat jalan dan komponen komponen yang terdapat didalamnya.

2.6.3. Diagram Perilaku

Diagram ini untuk memvisualisasi, menspesifikasi, membangun dan mendokumentasikan aspek dinamis dari sistem. Diagram perilaku di UML terdiri dari :

1. Diagram *use-case* (*Use case diagram*)

   Diagram ini menunjukkan sekumpulan kasus fungsional dan *actor* (jenis kelas khusus) dan keterhubungannya.

2. Diagram sekuen (*Sequence diagram*)

   Diagram ini menunjukkan interaksi yang terjadi antar objek. Diagram ini merupakan pandangan dinamis terhadap sistem.

3. Diagram kolaborasi (*Collaboration diagram*)

   Diagram ini juga merupakan diagram interaksi. Diagram ini menekankan pada organisasi struktur dari objek-objek yang mengirim dan menerima pesan.

4. Diagram *statechart* (*Statechart diagram*)

   Diagram ini adalah *state-machine* diagram, berisi state, transisi, kejadian dan aktivitas. Statechart merupakan pandangan dinamis dari sistem.

5. Diagram aktivitas (*Activity Diagram*)

   Diagram ini untuk menunjukkan aliran aktivitas di sistem. Diagram ini adalah pandangan dinamis terhadap sistem.

2.6.4. Notasi dalam UML (*Unified Modeling Language*)

UML menyediakan beberapa notasi dan artifak standar yang bisa digunakan sebagai alat komunikasi bagi para pelaku dalam proses analisis dan desain. Artifak di dalam UML didefinisikan sebagai informasi dalam berbagai bentuk yang digunakan atau dihasilkan dalam proses pengembangan perangkat lunak.

1. *Actor*

Actor adalah segala sesuatu yang berinteraksi dengan sistem aplikasi komputer. Jadi *actor* ini bisa berupa orang, perangkat keras, atau mungkin objek lain dalam sistem yang sama. Biasanya yang dilakukan *actor* adalah memberikan informasi pada sistem dan atau memerintahkan sistem untuk melakukan sesuatu.

2. *Class*

Class merupakan pembentuk utama dari sistem berorientasi objek karena *class* menunjukkan kumpulan objek yang memiliki atribut dan operasi

yang sama. *Class* digunakan untuk mengabstraksikan elemen-elemen dari sistem yang sedang dibangun.

3. *Interface*

Interface merupakan kumpulan operasi tanpa implementasi dari suatu *class*.

4. *Use Case*

Use case menjelaskan urutan kegiatan yang dilakukan *actor* dan sistem untuk mencapai suatu tujuan tertentu walaupun menjelaskan kegiatan. Namun *use case* hanya menjelaskan "apa" yang dilakukan oleh *actor* dan sistem, bukan "bagaimana" *actor* dan sistem melakukan kegiatan tersebut.

5. *Intercation*

Interaction digunakan untuk menunjukkan baik aliran pesan atau informasi antar obyek maupun hubungan antar objek.

6. *Package*

Package adalah kontainer atau wadah konseptual yang digunakan untuk mengelompokkan elemen-elemen dari sistem yang sedang di bangun, sehingga bisa dibuat model yang lebih sederhana. Tujuannya adalah untuk mempermudah penglihatan (*visibility*) dari model yang sedang dibangun.

7. *Note*

Note digunakan untuk memberikan keterangan dan komentar tambahan dari satu elemen sehingga bisa langsung terlampir dalam model. *Note* ini bisa ditampilkan ke semua elemen notasi yang lain.

8. *Dependency*

Dependency merupakan relasi yang menunjukkan bahwa perubahan pada salah satu elemen memberi pengaruh pada elemen lain.

9. *Boundary Class*

Boundary class adalah *class* yang menghubungkan *user* dengan sistem.

10. *Control Class*

Control class adalah *class* yang mengkoordinasi aktivitas dalam sistem. *Class* ini menghubungkan *boundary class* dengan *entity class.*

11. *Entity Class*

Entity Class adalah *class* yang menghubungkan dengan data atau informasi yang digunakan oleh sistem. *Entity class* ini adalah *class* yang menyimpan dan mengelola data dapat dikombinasikan.

2.7. *Tools* Pembangun

2.7.1. PHP Versi 4.0

Dunia internet semakin berkembang dalam penggunaannya dalam bidang komunikasi dan informasi, baik yang bersifat intern maupun umum. Yang dimaksud dengan informasi intern adalah data yang disimpan dalam *server* hanya dapat diakses oleh anak perusahaan atau kantor cabang dari sebuah perusahaan yang ada di luar daerah.

Data umum boleh di akses oleh semua pihak. Untuk membuat *website* yang dinamis dan mudah di *update* setiap saat dari *browser*, dibutuhkan sebuah program yang mampu mengolah data dari komputer *client* atau dari komputer

server itu sendiri sehingga mudah dan nyaman disajikan di *browser*. Salah satu program yang dapat dijalankan di *server* dan cukup handal adalah PHP. PHP adalah salah satu bahasa pemrograman yang berjalan dalam sebuah *web server* dan berfungsi sebagai pengolah data pada sebuah *server*. Dengan menggunakan program PHP, sebuah *website* akan lebih interaktif dan dinamis. Data yang dikirim oleh pengunjung *webiste*/komputer *client* akan diolah dan disimpan pada *database web server* dan dapat ditampilkan kembali apabila di akses. Untuk menjalankan kode-kode program PHP ini, file harus di *upload* ke dalam *server*.

PHP pertama kali dibuat oleh Rasmus Lerdorf pada tahun 1995. Pada versi pertamanya, PHP terdiri dari sekumpulan *script* PERL yang digunakan untuk mengolah data *form* dari *web*. Pada awalnya PHP bernama FI (*form intrepreted*). Setelah Rasmus melepaskan kode sumbernya, maka terbentuklah nama PHP/FI (*Personal Home Page/Form Intrepreted*). Sejak itulah PHP bersifat *open source*.

Pada bulan November 1997, PHP/FI versi 2.0 dirilis. Pada rilis ini *intrepreter* sudah diimplementasikan dalam bahasa C. Masih pada tahun yang sama sebuah perusahaan bernama Zend menulis ulang intrepreter PHP menjadi lebih bersih, lebih baik dan lebih cepat. Pada bulan Juni 1998, Zend kembali meriis *intrepreter* baru untuk PHP dan meresmikan nama rilis tersebut menjadi PHP 3.0.

Kemudian pada pertengahan tahun 1999, Zend merilis kembali *intrepreter* PHP baru dengan nama PHP 4.0. PHP 4.0. merupakan versi PHP yang paling populer di kalangan *programmer web*. Alasan yang menjadikan versi 4.0. begitu banyak diminati adalah kemampuannya untuk membagun aplikasi *web* yang

sangat kompleks, namun tetap stabil dalam kecepatan proses dan stabilitas yang tinggi.

PHP adalah salah satu bahasa *server-side* yang didesain khusus untuk aplikasi *web*. PHP dapat disisipkan diantara bahasa HTML dan karena bahasa *server-side*, maka bahasa PHP akan diekseskusi di *server*, sehingga yang dikirimkan ke *browser* adalah "hasil jadi" dalam bentuk HTML dan kode PHP tidak akan terlihat.

Saat ini banyak *website* yang menggunakan PHP sebagai dasar pengolahan data. Beberapa keunggulan yang dimiliki program PHP adalah :

1. PHP merupakan sebuah bahasa *script* yang tidak melakukan sebuah kompilasi dalam penggunaannya.
2. PHP memiliki tingkat akses yang lebih cepat.
3. PHP memiliki tingka *lifecycle* yang cepat sehingga mengikuti perkembangan teknologi internet.
4. PHP memiliki tingkat keamanan yang tinggi.
5. PHP mampu berjalan di beberapa *server* yang ada, misalnya Apache.
6. PHP juga mendukung akses ke beberapa *database* yang sudah ada, baik yang bersifat *free* ataupun komersil. *Database* tersebut antara lain MySQL, PostgreSQL, mSQL, dan MicrosoftSQL *serve*r.
7. PHP bersifat *free* atau gratis.

Salah satu faktor yang membuat PHP menjadi sangat populer sebagai bahasa *script* dalam pembuatan aplikasi berbasis *web* dan situs *web* dinamis adalah karena bahasa ini mendukung demikian banyak sistem *database*, mulai

dari mSQL, MySQL, MS-Access, PostgreSQL, bahkan sampai Oracle. Fungsi-fungsi untuk mempermudah pengaksesan berbagai jenis *database* tersebut tersedia lebih dari cukup pada PHP, sehingga meringankan, menyederhanakan, serta mempercepat proses pengembangan aplikasi berbasis *web*. Dari sekian banyak kombinasi PHP dan sistem *database* yang ada, terpilih pasangan PHP/MySQL sebagai yang terbaik, karena keduanya adalah produk-produk terbaik dari gerakan *open-source*. MySQL memberikan hasil yang optimal dari sisi kecepatan dan reliabilitas manajemen data. Saat ini, PHP secara *built-in* telah mendukung MySQL tanpa perlu modul tambahan.

2.7.2. *Database* MySQL versi 4.0.21

MySQL adalah merupakan perangkat lunak untuk sistem manajemen database (*Database Manajemen System*). Karena sifatnya yang *open-source* dan memiliki kemampuan menampung kapasitas yang sangat besar, maka MySQL menjadi database yang sangat populer. MySQL dapat dijalankan dalam 2 sistem operasi, yaitu Windows dan Linux.

Tabel 2.1. Perbedaan Fitur Tiap Versi dalam MySQL

Versi	Fitur
4.0	*Union*
4.1	*Sub Select, Store Procedure, Trigger*
4.2	*View*
4.0 atau 4.1	*Foreign Key Constraint*
4.1 atau 4.2	*Cursor*

2.7.3. *Web* Server Apache Friend

Web server adalah suatu *server* internet yang menggunakan protokol HTTP untuk melayani semua proses pentransferan data. *Web server* hingga saat ini merupakan *server* yang dapat dikatakan sebagai tulang punggung bagi semua pengguna internet. Hal ini dikarenakan *web server* bukan hanya bisa dapat melayani jenis data dalam bentuk *text*, akan tetapi juga dapat menampilkan format-format data dalam bentuk gambar, baik gambar dalam bentuk 2D atau 3D, suara juga dapat berinteraksi dalam dunia *wireless* internet dengan menjadikannya sebagai sebuah WAP (*Wireless Access Protocol*) *gateway* dan sebagainya.

Apache *web server* memiliki program pendukung cukup banyak yang dapat memberikan layanan yang cukup bagi penggunanya. Berikut ini adalah beberapa program pendukung dari Apache *web server* yaitu Kontrol Akses, PHP (*Personal Homepage*), SSI (*Server Side Include*).

2.8. Pengertian Pembelajaran Bahasa Inggris

Belajar bahasa adalah sebuah cara untuk mengenal sebuah bangsa. Menguasai suatu bahasa berarti mengenal bangsa tersebut. Di sinilah letak keunggulan orang yang menguasai bahasa bangsa lain, terutama bahasa internasional. Bahasa Inggris sudah menjadi alat komunikasi internasional.

Dengan semakin dikenalnya Bahasa Inggris dalam kehidupan sehari-hari masyarakat, maka semakin meningkat pula keinginan masyarakat untuk mempelajari Bahasa Inggris secara menyeluruh. Menguasai Bahasa Inggris berarti seseorang memiliki semacam, lisensi untuk menjelajah dunia. Untuk komunikasi

lisan orang bisa mencoba mempraktikan sedikit demi sedikit, tetapi untuk menguasai bahasa tulis, peran tata bahasa sangat penting. Tata Bahasa Inggris memiliki kekhasan dan keunikan dibanding dengan bahasa lain. *Tenses* adalah pola yang sedemikian banyak dan rumit membuat pembelajar Bahasa Indonesia mengalami kendala serius untuk menguasainya.

Dalam buku-buku Bahasa Inggris telah dijelaskan tentang tata bahasa (*grammar*), cara menerjemahkan (*translation*), cara membaca (*reading*) maupun percakapan (*conversation*). Tetapi masih banyak yang belum menyadari, di samping materi-materi di atas masih banyak bagian lain juga yang sangat penting dalam Bahasa Inggris, yaitu *Idiom*.

Dalam sebuah kalimat Bahasa Inggris, *Idiom* juga memegang peranan yang penting seperti yang lainnya. *Idiom* ini sering digunakan dalam penyusunan kalimat Bahasa Inggris, salah satunya dalam percakapan Bahasa Inggris. Oleh karena itu, kamus *Idiom* ini dibuat untuk memberikan bantuan kepada para pengguna untuk mengenal *Idiom* yang terdapat dalam Bahasa Inggris.