BAB 2
TINJAUAN PUSTAKA

2.1 Universitas Pendidikan Indonesia

2.1.1 Sejarah Universitas Pendidikan Indonesia

Universitas Pendidikan Indonesia didirikan pada tanggal 20 Oktober 1954 di Bandung, diresmikan oleh mentri pendidikan pengajaran Mr. Muhammad Yamin. Semula bernama Perguruan Tinggi Pendidikan Guru (PTPG), didirikan dengan latar belakang sejarah pertumbuhan bangsa yang menyadari bahwa upaya mendidik dan mencerdaskan bangsa merupakan bagian penting dalam mengisi kemerdekaan. Beberapa alas an didirikannya PTPG antara lain :

Pertama, setelah Indonesia mencapai kemerdekaannya, bangsa Indonesia sangat haus akan pendidikan. Kedua, perlunya disiapkan guru yang bermutu dan bertaraf universitas untuk meningkatkan kualitas pendidikan yang akan merintis terwujudnya masyarakan yang sejahtera.

Gedung Utama UPI bermula dari puing sebuah villa yang bernama villa isola, merupakan sebuah gedung bekas peninggalan masa sebelum Perang Dunia II. (Pada masa perjuangan melawan penjajah gedung ini pernah dijadikan markas para pejuang kemerdekaan). Puing puing itu dibangun kembali dan kemudian menjelma menjadi sebuah gedung bernama Bumi Siliwangi yang megah dengan gaya arsitekturnya yang asli.

Bumi Siliwangi menjadi tempat pertama kali bagi Pemuda mendapatkan gemblengan pendidikan guru dan tingkat universitas. Sebagai realisasi keputusan Mentri Pendidikan Pengajaran dan Kebudayaan Republik Indonesia (Nomor 35742 tanggal 1 September 1954 tentang pendirian PTPG/Perguruan Tinggi Pendidikan Guru)

PTPG dipimpin oleh seorang dekan yang membawahi beberapa jurusan dan atau balai yakni :

- Ilmu Pendidikan
- Ilmu Pendidikan Jasmani

- Bahasa dan Kesusastaan Indonesia
- Bahasa dan Kesusastraaan Inggris
- Sejarah Budaya
- Pasti Alam
- Ekonomi dan Hukum Negara
- Balai Penelitian Pendidikan

Sejalan dengan keputusan Mentri Pendidikan, Pengajaran dan Kebudayaan No. 40718/S pada saat itu, yang menyatakan bahwa PTPG dapat berdiri menjadi perguruan tinggi dalam universitas, maka seiring dengan berdirinya Universitas Padjajaran (UNPAD), maka pada tanggal 25 November 1958 PTPG diintegrasikan menjadi fakultas utama Universitas Padjajaran dengan nama Fakultas Keguruan dan Ilmu Pendidikan (FKIP)

Untuk memantapkan sistem pengadaan tenaga guru dan tenaga kependidikan, berbagai kursus yang ada pada saat itu, yaitu pendidikan guru BI dan BII. Diintegrasikan kedalam FKIP melalui Surat Keputusan Mentri Pendidikan Dan Kebudayaan No. 6 tahun 1961. Selanjutnya FKIP Berkembang menjadi FKIP A dan FKIP B. pada saat yang sama, berdiri pula Institut Pendidikan Guru (IPG), yang mengakibatkan adanya dualisme dalam Lembaga Pendidikan Guru. Untuk menghilangkan dualisme tersebut, pada tanggal 1 Mei 1963, yang melebur FKIP dan IPG menjadi Institut Keguruan dan Ilmu Pendidikan (IKIP) sebagai satu satunya lembaga pendidikan guru tingkat Universitas. FKIP A, FKIP B dan IPG yang ada di Bandung akhirnya menjadi Institut Keguruan dan Ilmu Pendidikan Bandung (IKIP Bandung).

IKIP bandung pada saat itu telah memiliki lima fakultas, yaitu Fakultas Ilmu Pendidikan, Fakultas Keguruan Ilmu Sosial, Fakultas Keguruan Sastra dan Seni, Fakultas Keguruan Ilmu Eksakta dan Fakultas Keguruan Ilmu Teknik. Kebutuhan akan tenaga guru kian mendesak, demikian pula tumbuhnya hasrat untuk meningkatkan dan memeratakan kemampuan pada guru. Hal ini mendorong IKIP Bandung membuka ekstension, antara tahun 1967-1970 IKIP bandung membuka ekstension hamper di seluruh kabupaten di Jawa Barat.

Peranan IKIP Bandung di Tingkat Nasional semakin Menonjol, setelah pemerintah menetapkan bahwa Ikip Bandung menjadi IKIP Pembina yang diserahi tugas membina IKIP di luar pulau Jawa, yaitu IKIP cabang Banda Aceh, Palembang, Palangkaraya dan Banjarmasin. Sesuai dengan kebijaksanaan Departemen Pendidikan dan Kebudayaan pada awal tahun 1970an secara bertahap ekstension tersebut ditutup dan cabang cabang IKIP di daerah menjadi Fakultas di lingkungan daerah masing-masing.

Untuk meningkatkan mutu pengajar, pada tahun 1970 IKIP Bandung membuka Program Pos Doktoral melalui pembentukan Lembaga Pendidikan Pos Doktoral (LPPD) PPS yang mengelola Program S2 dan S3. Pada tahun 1976 LPPD berubah menjadi Fakultas Sekolah Paska Sarjana dan pada tahun 1991 menjadi Program Paska Sarjana.

Penataan Program Pendidikan tinggin yang dilakukan oleh pemerintah dengan menerapkan multiprogram dan multistrata, ditindaklanjuti IKIP Bandung dengan membuka Program Diploma Kependidikan, untuk meningkatkan kualifikasi guru SD menjadi lulousan DII, tahun ajaran 1990/1991. Diselenggarakannya Diploma DII Pendidikan Guru Sekolah Dasar. Selain diselenggarakannya di Kampus Bumi Siliwangi Program ini juga diselenggarakan di Unit Pelaksana Program (UUP) pada beberapa sekolah eks SPG yang diintegrasikan ke IKIP. Guna meningkatkan kualifikasi Guru Taman Kanak-kanak atau Play Group pada tahun 1996/1997 IKIP Bandung membuka Program DII PTGK.

Seiring dengan kebijakan Pemerinatah dibidang pendidikan tinggi yang memberikan perluasan mandate bagi Lembaga Pendidikan Tenaga Kependidikan (LPTK) yang harus memenuhi tuntutan perubahan serta mengantisipasi segala kemungkinan dimasa yang akan datang, IKIP Bandung diubah menjadi Universitas Pendidikan Indonesia melalui keputusan Presiden RI No.124 tahun 1999 tertanggal 7 Oktober 1999.

Untuk memperluas jangkauan dalam mendukung pembangunan nasional, UPI harus mampu berdiri sendiri dan berkiprah. Kebulatan tekad ini menumbuhkan keyakinan akan kemampuan yang telah dimilikinya. Tekad ini

memberi keyakinan kepada Pemerintah bahwa UPI telah dapat berdiri sendiri dan dapat diberikan tanggung jawab yang lebih besar. Dengan kepercayaan ini, melalui surat Pemerintah No. 6 tahun 2004 UPI diberi Otonomi dan menjadi Perguruan Tinggi Badan Hukum Milik Negara (PT BHMN)

Pengembangan dan peningkatan UPI tidak saja berorientasi pada bidang akademik, tetapi juga dalam berbagai bidang termasuk pemantapan konsep dan rencana pembangunannya. Melalui bantuan Islamic Development Bank (IDB) telah merancang dan menata pembangunan gedung kampus yang megah, modern dan representative sebagai penunjang kegiatan belajar mengajar bermodalkan kemampuan yang dimiliki. Universitas Pendidikan Indonesia bertekad menjadikan lembaga pendidikan ini terdepan dan menjadi Universitas Pelopor dan Unggul.

2.1.2 Perpustakaan Universitas Pendidikan Indonesia

Perpustakaan UPI lahir pada tanggal 20 Oktober 1954 dibawah naungan Perguruan Tinggi Pendidikan Guru (PTPG), di hotel Homann Jl. Asia Afrika Bandung tahun 1954. Perpustakaan pada tahun 1954 memiliki koleksi berjumlah 8000 eksemplar untuk melayani kebutuhan 214 mahasiswa yang tergabung kedalam 7 jurusan. Lokasi yang berada dihotel dirasa kurang sesuai untuk kegiatan akademik, sehingga lokasi dipindahkan ke PPI Lembang, dengan luas ruangan 195 m2. Pada waktu itu perpustakaan memiliki memiliki koleksi berjumlah sekitar 25000 judul buku.

Tahun 1963 perpustakaan dipindahkan ke gedung baru II (gedung FPBS) yang luasnya 480 m2. Pada tahun 1969 perpustakaan dipindahkan lagi ke gedung garnadi dengan menempati lahan seluas 750 m2.

Dengan semakin pesatnya perkembangan jumlah mahasiswa dan bertambahnya jumlah jurusan, maka diperlukan sumber informasi yang lebih lengkap serta ruang yang lebih luas. Oleh karena itu, dibangunlah sebuah gedung baru seluas 1200 m2, gedung ini bertahan dari tahun 1979 sampai 1994, sampai akhirnya para pembuat kebijakan membuat gedung baru yang lebih lebih besar dari gedung yang sudah ada. Maka mulai bulan April 1995, resmilah perpustakaan IKIP Bandung. Menempati gedung baru seluas 12000 m2, yang menurut

rancangannya akan mampu menampung koleksi total sejumlah 1.000.000 eksemplar, serta menerima pengunjung sejumlah 5000 orang setiap harinya

.

2.2 Landasan Teori

2.2.1 Database

Database adalah merupakan kumpulan informasi bermanfaat yang diorganisasikan kedalam sebuah aturan yang khusus. Informasi ini adalah data yang telah diorganisasikan ke dalam bentuk yang sesuai dengan kebutuhan seseorang. [3]

Menurut Ramez Elmasri [4] mendefinisikan basis data lebih dibatasi pada arti implicit yang khusus, yaitu:

- Basis Data merupakan penyajian suatu aspek dari dunia nyata (real world)
- Basis data merupakan kumpulan data dari berbagai sumber yang secara logika mempunyai arti implisit, sehingga data yang terkumpul secara acak dan tanpa mempunyai arti tidak dapat disebut basis data.
- Basis data perlu dirancang, dibangun dan data dikumpulkan untuk suatu tujuan. Basis data dapat digunakan oleh beberapa user dan beberapa aplikasi yang sesuai dengan kepentingan user.

Dari beberapa definisi tersebut, dapat dikatakan bahwa basis data mempunyai berbagai sumber data dalam pengumpulan data, bervarisi derajat interaksi kejadian dari dunia nyata, dirancang dan dibangun agar dapat digunakan oleh beberapa user untuk berbagai kepentingan [5]

Gabungan antara basis data dan aplikasi perangkat lunak DBMS (*Database Management System*) termasuk didalamnya program yang dibuat dan bekerja dalam satu sistem disebut dengan Sistem basis data [5]. Sistem basis data dapat dianggap sebagai tempat untuk sekumpulan berkas data yang terkomputerisasi dengan tujuan untuk memelihara informasi dan membuat informasi tersebut tersedia saat dibutuhkan.

2.2.2 NoSQL

NoSQL adalah sistem managemen basis data non-relational, pengambilan informasi yang lebih cepat dari database dan portable. NoSQL pada dasarnya berasal dari sistem database RDB. Database ini biasanya berinteraksi dengan sistem operasi UNIX. Database NoSQL adalah mereka yang non-relational, open source, didiatribusikan di dunia serta memiliki kinerja tinggi dengan cara linear yang terukur. Database non-relational tidak mengatur data dalam tabel terkait. Database NoSQL adalah open source, sehingga setiap orang dapat melihat source code secara bebas, memperbaharuinya sesuai dengan kebutuhan dan kompilasi.

NoSQL dikembangkan pertama kali pada tahun 1998 oleh Carlo Strozi, kemudian pada tahun 2009, Eric Evans memperkenalkan kembali NoSQL.

2.2.2.1 Konsep NoSQL

Menurut Eric Brewer [6], NoSQL didasarkan pada teori *Consistency, Availability, Partition-Tolerance* (CAP). *Consistency Availability* (CA) berseberangan dengan *Partition-Tolerance* dan berhubungan dengan replikasi. *Consistency Partition- Tolerance* (CP) berseberangan dengan *Availability* dalam penyimpanan data. *Availability Partition-Tolerance* (AP) dimana system mencapai kondisi *eventual consistency* melalui replikasi dan verifikasi yang konsisten dalam *node* yang telah terbagi – bagi. Dalam basis data NoSQL, penerapan konsep tersebut diterjemahkan dalam empat konsep dasar, yaitu *Non-Relational, MapReduce, Schema Free,* dan *Horizontal Scaling.*

1) Non-Relational

Konsep *Non-Relational* dalam basis data *NoSQL* meliputi hirarki, graf, dan basis data berorientasi obyek yang sudah terlebih dahulu ada sejak tahun 1960 sebelum akhirnya basis data relasional muncul pada tahun 1970 [7]. Penggunaan basis data non-relational kembali merebak seiring dengan bertambahnya aplikasi berbasis web yang memerlukan banyak penyimpanan data. Meskipun memiliki kelemahan pada konsistensi dan redundansi data, namun basis data non-relasional dapat menyelesaikan beberapa permasalahan terkait dengan *avaibility,* dan *partition tolerance*. Tugas pengecekan konsistensi dan redundansi data diserahkan

pada sisi aplikasi, sedangkan basis data nonrelational hanya bertugas memanipulasi penyimpanan saja.

2) MapReduce

MapReduce merupakan model pemrograman yang diadaptasi dari pemrograman fungsional yang diimplementasikan untuk mengolah *dataset* yang sangat besar. Tujuan dari *MapReduce* adalah merancang suatu abstraksi baru yang memungkinkan pengguna untuk membuat antarmuka pemrograman sederhana dan menyembunyikan detail yang rumit dari pararelisasi, *fault-tolerance*, distribusi data, dan *load balancing* dalam pustaka pemrogramannya. Hasilnya menunjukkan bahwa penerapan *MapReduce* dapat menyederhanakan antarmuka pemrograman yang dapat mendukung paralelisasi dan distribusi komputasi skala besar secara otomatis [8]

Pemrograman dengan *MapReduce* telah sukses diterapkan penggunaannya oleh *Google* untuk berbagai tujuan, salah satunya adalah *Google indexing*. Dalam kinerjanya, *Google* menerapakan ribuan mesin yang bekerja pada ratusan *terabytes* data dengan lokasi server yang tersebar di beberapa lokasi. Jenis arsitektur seperti ini dapat mengurangi waktu kinerja yang diperlukan. Pembangunan arsitektur *Google* dengan menggunakan *MapReduce* memerlukan waktu hanya dalam beberapa jam saja dibandingkan dengan tidak menerapkan *MapReduce* yang memerlukan waktu selama berbulan bulan. Penggunaan *library* dalam *MapReduce* memiliki beberapa keuntungan seperti *load balancing*, optimasi perangkat penyimpanan yang nantinya akan meningkatkan keefisiensian sistem dan kemudahan dalam penggunaannya. Mudahnya, *MapReduce* bekerja dengan membagi proses menjadi dua fase, yaitu tahap *map* dan tahap *reduce*. Seorang *programmer* dapat memanfaatkan dua fungsi ini bersama fungsi *key-value pairs* sebagai *input* dan *output* untuk mencapai semua fase [9]

3) Schema-free

NoSQL dan *RDBMS* mempunyai perbedaan dalam hal penerapan skema basis data. Dalam basis data relasional, sebuah tabel didesain dengan peraturan skema yang ketat. *NoSQL* menyimpan data dengan aturan yang lebih longgar, artinya tidak seperti basis data yang berdasarkan *SQL* tradisional , *NoSQL* tidak memiliki

tabel, kolom, *primary* dan *foreign key*, join, dan relasi [10]. Dalam pengembangan basis data relasional, *developer/database administrator* harus berhati-hati dalam menentukan bagaimana tabel saling berelasi dan *field* yang ada didalam setiap tabel. Karena perubahan skema dalam RDBMS dapat menimbulkan masalah ketergantungan dan integritas, seperti timbulnya kolom *null* dan relasi kunci yang tidak cocok. Hal ini bukan masalah dalam *NoSQL* karena adanya penerapan *schemafree*. Setiap dokumen bertanggung jawab terhadap isinya sendiri, maksutnya *null value* dapat dihilangkan dalam beberapa baris, dan *field* baru dapat didefinisikan dalam setiap dokumen secara indpenden [10]. Salah satu produk basis data *NoSQL* yang menerapkan fitur *schema-free* adalah *Cassandra*. Dalam *Cassandra,* pengembang hanya perlu mendefinisikan *keyspace* sebagai pertimbangan *container* dan *contains column families*. *Keyspace* hanya digunakan sebagai *logical namespace* yang dapat dimasukkan dalam *configurations* dan *hold column families*. *Column families* artinya sebuah nama untuk relasi data dan hirarki kolom (memungkinkan penyisipin tabel dalam kolom). Disamping itu, kita hanya perlu menambahkan data dalam tabel, menggunakan kolom, tanpa menentukan kolomnya terlebih dahulu. Dalam *Cassandra* kita hanya perlu mendefinisikan tabel *namespace* dan hirarki kolom didalamnya, memodelkan dan mendesain tabel join adalah masalah dalam basis data relasional bukan NoSQL.

Manfaat lain dalam penggunaan *schema-free* adalah penghematan dalam media penyimpanan. Dalam basis data relasional, setiap *field* yang ada dalam tabel harus mempunyai nilai, walaupun nilai itu *null*. Model data *schema free* artinya setiap baris memungkinkan memiliki nilai sebanyak yang telah didefinisikan dalam tiap *fields*, dan tidak perlu menggunakan nilai yang memang tidak diperlukan. Kelemahan dalam penggunaan *schema free* adalah memunculkan lemahnya pendefinisian struktur yang menungkinkan terjadinya penggunaan basis data yang tidak konsisten. Jika tujuan pembangunan basis data didasarkan pada konsistenti yang ketat, seperti *wiki*, *document management systems, discussion forums, blogs*, dan *support management systems*, basis data relasional masih merupakan pilihan yang tepat [10].

4) Horizontal Scaling

Horizontal scaling memungkinkan basis data dijalankan pada beberapa server untuk meningkatkan kemampuan perangkat penyimpanan dan meningkatkan efisiensi waktu. Hal ini memerlukan kemampuan dinamis pemartisian data dalam serangkaian *node* (seperti *storage hosts*) dalam suatu *cluster server*. Kemampuan untuk meningkatkan kemampuan dengan menambahkan beberapa komputer sangatlah penting dilakukan untuk data yang jumlahnya banyak, karena *vertical scaling* dilakukan dengan meningkatkan kemampuan spesifikasi *single server* (misalnya penambahan prosesor, memori, dan peralatan penyimpanan) terbatas untuk dilakukan dan lebih mahal [11].

Horizontal scaling berarti memungkinkan dilakukannya penambahan server dalam satu jaringan dan *user* tidak sadar jika ada *hardware* yang diganti dari sisi server (*transparent*). Ada beberapa teknik partisi yang digunakan dalam basis data untuk melakukan *horizontal scaling*, salah satunya adalah *consistent hashing* yang digunakan dalam *Cassandra* dan *Amazon Dynamo* Kunci dari menerapkan *consistent hashing* adalah membuat suatu lingkaran atau "*ring*". Setiap *node* dalam sistem yang ditandai dengan *random value* dalam suatu *space* yang merepresentasikan posisi dari *ring*. Suatu kunci *item* ditandai dengan sebuah *node* untuk memperoleh posisinya pada *ring*, kemudian berpindah menuju *node* selanjutya yang sudah ditandai. *Node* memerankan peranan penting sebagai koordinator untuk kunci yang akan digunakan dalam *route request*. Kemudian, setiap *node* menjadi tanggung jawab dari daerah yang ada di *ring* diantara *noder* dan *node* dalam *ring* sebelumnya. Pejelasan detail mengenal *Dynamo* dijelaskan oleh DeCandia dkk [11].

2.2.2.2 Model Data NoSQL

NoSQL merupakan basis data non relasional dengan *schema-free* yang memunculkan pertanyaan bagaimana *NoSQL* dapat melakukan partisi untuk data yang berukuran besar, melakukan *query*, replikasi data, dan mendukung adanya konsistensi. Bagian ini berisi penjelasan empat model data *NoSQL*, yaitu *column-oriented, document-oriented, object-oriented* dan *graph-oriented* [6].

1) Column-Oriented

Penerepan *column-oriented* terdapat dalam *Casssandra*. *Cassandra* menggunakan distribusi multidimensional *map indexed* dengan sebuah *key*. Baris kunci yang sering digunakan dalam *Cassandra* adalah *string* dengan panjang 16-36 *bytes*. Setiap kolom digabungkan menjadi sebuah *coloumn families.* Wei Kang [6] menerapkan beberapa konsep dalam penerapan *column-oriented*. Suatu kolom unit *atomic* dari informasi didukung oleh *Cassandra* yang diekspresikan dengan nama *value*. *Super-column* merupakan gabungan dari kolom dengan nama yang umum dan digunakan untuk pemodelan tipe data yang komplek. Baris secara unik mengidentifikasikan data yang terdapat dalam *column* dan *super-column*. Dalam *Cassandra* baris dapat dikenali dengan sebuah kunci. *Column Family* merupakan bagian dari suatu unit abstraksi yang berisi baris kunci yang tergabung dalam *column* dan *super column* yang memiliki struktur data yang tinggi. *Keyspace* merupakan level teritnggi dari unit informasi yang terdapat dalam *Cassandra*. Kumpulan *column families* sebenarnya merupakan subordinat dari satu *keyspace*. Pada intinya model data *column-oriented* memungkinkan suatu aplikasi secara bebas untuk mengembangkan bagaimana informasi disusun berdasarkan suatu desain *schema*.

2) Document-Oriented

Database berorientasi dokumen adalah sebuah program komputer yang dirancang untuk menyimpan, mengambil, dan mengelola informasi berorientasi dokumen, juga dikenal sebagai data semi-terstruktur. Database berorientasi dokumen adalah salah satu kategori utama database NoSQL dan popularitas istilah "database dokumen-berorientasi" (atau "toko dokumen") telah berkembang dengan penggunaan istilah NoSQL itu sendiri. Berbeda dengan database relasional dan gagasan mereka "Tabel" dan "Hubungan" (antara Tabel), sistem ini dirancang di sekitar gagasan abstrak dari "Dokumen".

Contoh penerapan model data *document-oriented* terdapat dalam *CouchDB* yang dibangun oleh *IBMTM*. Basis data yang memiliki model data *document-oriented* sangat bermanfaat untuk suatu domain yang bentuk masukannya dokumen yang tidak terstruktur seperti *web pages, wikis, discussion forums,* dan *blogs*. Data tersimpan dalam basis data *CouchDB* yang mencakup serangkaian

dokumentasi yang berisi beberapa atribut dan nilai dengan masing-masing *id* yang unik dan *metadata*. *CouchDB* tidak pernah melakukan *overwrite document*, melainkan menambahkan dokumen baru ke basis data bila diperlukan seperti ketika terjadi proses *update*.[6]

3) Object-Oriented

Basis data berorientasi objek adalah model basis data dimana informasi direpresentasikan dalam bentuk objek yang digunakan dalam pemrograman berorientasi obyek. *OrientDB* adalah contoh dari basis data *object-oriented*. Dokumen *OrientDB* yang ada dalam suatu *cluster*, dapat berupa fisik, logis atau *in-memory*, yang digunakan untuk menyimpan *link* ke dalam data. *Cluster* adalah cara yang sangat umum untuk mengelompokkan *record*, hal ini merupakan suatu konsep yang tidak ada dalam basis data relasional. Cara ini dapat mengelompokkan semua *record* pada jenis tertentu, atau dengan nilai-nilai tertentu. *OrientDB* menggunakan segmen data untuk menyimpan isi *record*. Segmen data mirip dengan *file physical cluster* yang menggunakan dua atau lebih *file*, yaitu satu atau beberapa *file* dengan ekstensi "*oda*" (*Orient Data*) dan hanya satu *file* dengan ekstensi "*odh*" (*Orient data Holes*) [16].

4) Graph-Oriented

Basis data grafik (*GraphDB*) adalah basis data yang menggunakan struktur grafik yang berisi *node*, *edge*, dan property untuk mewakili dan menyimpan informasi. *GraphDB* diperlukan untuk data grafik yang berskala besar, terutama yang dipergunakan oleh para peneliti biologi jaringan dan situs jaringan sosial, seperti *Facebook*, dan *Twitter*. *GraphDB* memetakan secara langsung objek ke aplikasi dan lebih intuitif untuk menggambarkan *data set* asosiatif. Beberapa keuntungan dari *GraphDB* adalah [16]: *Intuitive*, dimengerti oleh pikiran manusia, yaitu menggambarkan entitas dan hubungan sebagai grafik masalah umum yang akrab dengan manusia; *Elemental* untuk ilmu komputer, yaitu grafik, terutama grafik pohon (seperti *binary-tree, B+ tree, red-black tree*) berfungsi sebagai struktur data dasar dalam ilmu komputer dan berbagai masalah (*shortest path* dan *max-flow*) dapat diubah dan diselesaikan dengan algoritma grafik; *Ubiquitous*,

yaitu pemodelan *ER* ke model jejaring sosial selalu dikelilingi oleh grafik baik di computer ataupun dalam kenyataan.

2.2.3 MongoBD

MongoDB adalah dokumen database yang memberikan kinerja tinggi *(high performance)*, ketersediaan tinggi *(high avability)* dan skalabilitas yang mudah *(easy scalability)*.

2.2.3.1 Tipe Data

Format penulisan pada MongoDB lebih dikenal dengan nama BSON (Binary JSON). Ada beberapa keuntungan BSON daripada JSON, diantaranya adalah JSON tidak dapat menerima data dengan tipe date, sedangkan BSON dapat. JSON tidak dapat membedakan data integer, double dan float, dia hanya memiliki data dengan tipe data number (Double-precision floating-point format) sedangkan BSON dapat membedakannya, selain itu juga ada beberapa tipe data seperti symbol, regex, dan code yang tidak ada dalam JSON tapi ada di BSON. Tipe data yang di *support* oleh MongoDB :

1) *Null* : Tipe data ini untuk merepresentasikan null *value* atau ketidaksediaan key sebuah data.
2) *Boolean* : Hanya ada dua nilai dari boolean, *true* atau *false*
3) 64-bit floating point number : Pada shell mongodb, semua number akan di ubah ke bentuk ini 64-bit floating point number.
4) *String* : *String* mongodb sudah menggunakan encoding unicode atau UTF8.
5) *Object id* : *Objek id* yang pasti ada pada tiap document, berukuran 12-byte dan akan digenerate otomatis saat pembuatan document.
6) date
7) *regular expression*: Sebuah key document juga bisa berisi regular expression (regex).
8) *Code* : Sebuah key document juga dapat menyimpan kode-kode javascript.
9) binary data: Tipe data ini tidak dapat dimanipulasi oleh *javascript* shell.

10) *Array* : Merupakan kumpulan dari beberapa value, tiap value dalam satu array boleh berbeda tipe datanya.
11) *embedded document* : Tipe data ini memiliki arti dokumen yang memiliki atau embbed *document* lain.

2.2.3.2 Pengelolaan Data

1) Pendekatan *Replication*

MongoDB mendukung replikasi, selain itu konfigurasi replikasi pada MongoDB sangatlah mudah jika dibandingkan sistem basis data RDBMS (IMHO). Hal ini membuat pemilik sistem yang memiliki traffic membaca tinggi (seperti portal berita, forum dan blog) akan sangat terbantu ketika sistemnya akan menggunakan replikasi.

Adanya database NoSQL seperti MongoDB yang mencoba untuk menyelesaikan permasalahan ini. Disini, Data tidak ditulis/dibaca dari database dengan menggunakan bahasa SQL, tetapi menggunakan metode object-oriented yang lebih disukai oleh developers. Kelebihan lainnya adalah adanya dukungan adanya banyaknya tipe index yang berbeda beda untuk lookupsterhadap data tertentu. Mereka juga memiliki kemampuan clustering secara default.

2) *Sharding*

Sharding adalah metode untuk menyimpan data pada beberapa mesin yang berbeda. MongoDB menggunakan *sharding* untuk mendukung *deployment* pada *data set* yang sangat besar dan operasi dengan *throughput* yang tinggi.

3) *Security and Authentication*

Didalam pengembangan mongoDB tidak memiliki solusi cepat untuk mengembangkan keamanan dan mengurangi resiko keamanan yang ada. Dokumendokumen di bagian ini menggunakan pendekatan “Pertahanan pada Kedalaman” untuk mengamankan *deployment* MongoDB dan alamat-alamat sejumlah metode yang berbeda untuk mengelola resiko dan mengurangi pembukaan resiko. Maksud dari pendekatan “Pertahanan pada

Kedalaman" adalah untuk memastikan tidak ada poin kegagalan yang dapat dieksploitasi dalam *deployment*, yang dapat memungkinkan penyusup atau pihak yang tidak dapat dipercaya untuk mengakses data yang tersimpan dalam *database* MongoDB.

2.2.4 HTML

Sebuah file HTML merupakan file teks biasa yang mengandung tag-tag HTML. Karena merupakan file teks, maka HTML dapat dibuat dengan menggunakan teks editor yang sederhana, misalnya Notepad. Dapat juga menggunakan HTML editor yang bersifat Visual, misalnya Frontpage.

Untuk menandai bahwa sebuah file teks merupakan file HTML, maka cirri yang paling nampak jelas adalah ekstensi filenya, yaitu .htm atau .html.

StrukturDasar

```
<HTML>
<HEAD>
……
</HEAD>

<BODY>
……
</BODY>
</HTML>
```

Bagian Head sebenarnya tidak harus ada pada dokumen HTML, tetapi pemakaian head yang benarakan meningkatkan kegunaan suatu dokumen HTML. Isi bagianhead-kecuali judul dokumen-tidak akan terlihat oleh pembaca dokumen tersebut.

Elemen-elemen pada bagian head akan mengerjakan tugas-tugas sebagai berikut:

- Menyediakan judul dokumen
- Menjembatani hubungan antar dokumen
- Memberitahu browser untuk membuat form pencarian
- Menyediakan metode untuk mengirim pesan ke tipe browser

Bagian BODY merupakan isi dari dokumen HTML. Semuain formasi yang

akan ditampilkan, mulai dari teks, gambar, sound, dan lain-lain, akan ditempatkan di bagian ini. Seperti telah disebutkan diatas, bagian BODY diawali oleh tag<BODY> dan ditutup </BODY>.

Untuk mempermudah pembacaan kembali kode-kode HTML, kadang-kadang ditambahkan komentar kedalam sebuah dokumen. Agar komentar tidak dapat dibaca pada browser, maka harus digunakan tanda khusus, yaitu <!—dan diakhiri dengan tanda-->

2.2.5 CSS

CSS (*Cascading Style Sheet*) adalah sebuah cara untuk memisahkan isi dengan layout dalam halaman-halaman web, CSS digunakan untuk mempersingkat penulisan tag HTML seperti font, color, text dan table menjadi lebih ringkas sehingga tidak terjadi perulangan tulisan. CSS mengendalikan beberapa komponen dalam sebuah web sehingga akan lebih terstruktur dan seragam. Penggunaan CSS dalam sebuah web akan lebih fleksibel dalam penampilannya dan juga akan meminimalkan penggunaan tag yang berpengaruh kedalam ukuran file. Meskipun CSS telah didukung oleh kebanyakan browser saat ini, penerjemah tiap browser untuk menampilkan CSS akan berbeda-beda.

Dalam pembuatan document web menggunakan style CSS dikenal adanya aturan atau struktur penulisan baku agar style css yang dibuat dapat tampil dengan baik dalam document web. Strukturnya adalah

```
<html>
<head><title>….</title>
<style type=”text/css”>
</syle></head>
<body></body>
</html>
```

Selector adalah nama-nama yang diberikan untuk syle yang berbeda, baik itu style internal, maupun external.Karena CSS memiliki aturan tersendiri dalam

penulisan sintaknya, penulisan sintak tersebut terbagi kedalam tiga bagian.Yaitu selector, property dan value.

Ada beberapa teknik cara penempatan CSS dalam HTML, yaitu :

1. *Inline Style*

   Inline syle adalah CSS yang dibuat dalam sebuah tag HTML yang hanya berlaku untuk document yang diapitnya saja. Teknik ini digunakan pada pemformatan khusus pada sebuah element dalam document web. Penulisannya cukup hanya menambahkan atribut style dalam tag HTML tersebut.

2. *Internal Style*

   Internal style adalah CSS yang dibuat dan diletakan dalam HTML, pada teknik ini style diletakan pada tengah tag antara tag <head> dan </head>, aturan dalam style ini diatur sedemikian rupa untuk digunakan pada suatu tempat maupun untuk keseluruhan situs.

3. *External Syle*

   Suatu teknik yang digunakan untuk memanggil style CSS pada file CSS yang menggunakan perintah “link rel” yang berfungsi untuk menghubungkan kedalam sebuah style CSS external dengan nama yang sudah ditentukan.

   Contoh: <link rel=”stylesheet” type=”text/CSS” href=”default.css”

   Dari contoh tersebut, menghubungkan sebuah document HTML dengan sebuah style external dengan nama default.css yang telah dibuat sebelumnya, sehingga style tersebut dihubungkan untuk memformat tampilan dalam dokumen HTML tersebut

2.2.6 PHP

PHP dibuat oleh Rasmus Lerdorf pada tahun 1994, pada awalnya tidak untuk didistribusikan dan hanya digunakan pada homepage pribadinya. Pada tahun 1995 dikeluarkan versi pertama yang dapat digunakan oleh umum dengan nama Personal Home Page Tools. Ditulis kembali pada pertengahan-1995 dan diberi

nama sebagai PHP/FI Version 2. FI berasal dari paket Rasmus yang mana merupakan html interpreter untuk data form. Pada hasil kombinasi tersebut juga ditambah dukungan terhadap mSQL.PHP/FI terus berkembang dan banyak orang mulai memberikan kontribusi dalam pengembangannya.

Pada tahun 1996 PHP/FI diperkirakan telah digunakan 15,000 situs web didunia, dan pada pertengahan 1997 jumlah ini berkembang melebihi 50,000.Pada pertengahan 1997 juga terjadi perubahan pada PHP dimana berubah menjadi proyek yang didukung oleh team yang lebih terorganisasi.Parsernya ditulis ulang oleh Zeev Suraski dan Andi Gutmans dan parser baru inilah yang membentuk basis untuk PHP versi 3.Banyak kode utility dari PHP/FI yang di masukkan ke PHP dan banyak diantaranya telah selesai ditulis kembali.

Sekarang baik PHP/FI atau PHP telah diikutsertakan dalam sejumlah produk komersil seperti C2's StrongHold web server dan RedHat Linux.Suatu perkiraan yang konservatif didasarkan dari hasil ekstrapolasi terhadap angka yang diperoleh dari NetCraft, PHP diperkirakan telah digunakan oleh lebih dari 150,000 situs diseluruh dunia.Dan secara perspektif, angka ini lebih besar dibandingkan dengan server yang menjalankan Netscape's flagship Enterprise server di Internet.

PHP diawali dengan <? Dan di akhiri dengan ?>

```
<?php
echo "Hello, Nama saya PHP ";
?>
```

2.2.7 Analisis Berorientasi Objek

Analisis dan disain berorientasi objek adalah cara baru dalam memikirkan suatu masalah dengan menggunakan model yang dibuat menurut konsep sekitar dunia nyata. Dasar pembuatan adalah objek, yang merupakan kombinasi antara struktur data dan perilaku dalam satu entitas.

Pengertian "berorientasi objek" berarti bahwa kita mengorganisasi perangkat lunak sebagai kumpulan dari objek tertentu yang memiliki struktur data dan perilakunya.

2.2.7.1 Karakteristik Objek

1) Objek

- Identitas berarti bahwa data diukur mempunyai nilai tertentu yang membedakan entitas disebut Objek.
- Objek dapat kongkrit, seperti halnya arsip dalam sistem, atau konseptual seperti kebijakan penjadualan dalam multiprocessing pada sistem operasi.
- Setiap objek mempunyai sifat yang melekat pada identitasnya.
- Dua objek dapat berbeda walaupun bila semua nilai atributnya identik.

2) Klass

- Kelas merupakan gambaran sekumpulan Objek yang terbagi dalam atribut, operasi, metode, hubungan, dan makna yang sama.
- Suatu kegiatan mengumpulkan data (atribut) dan perilaku (operasi) yang mempunyai struktur data sama ke dalam satu grup.
- Kelas Objek merupakan wadah bagi Objek. Dapat digunakan untuk menciptakan Objek.
- Objek mewakili fakta/keterangan dari sebuah kelas

2.2.7.2 Unified Modeling Language

Unified Modeling Language (UML) adalah sebuah bahasa yang berdasarkan grafik/gambar untuk memvisualisasikan, menspesifikasikan, membangun dan pendokumentasian dari sebuah sistem pengembangan software berbasis *Object Oriented* (OO)

1) Hubungan / *Relationship*

- *Dependency*, adalah hubungan semantik antara dua benda/things yang mana sebuah benda berubah mengakibatkan benda satunya akan berubah pula. Umumnya sebuah dependency digambarkan sebuah panah dengan garis terputus-putus.
- Association, hubungan antar benda struktural yang terhubung diantara obyek. Kesatuan obyek yang terhubung merupakan hubungan khusus, yang menggambarkan sebuah hubungan struktural diantara seluruh atau

sebagian. Umumnya association digambarkan dengan sebuah garis yang dilengkapi dengan sebuah label, nama, dan status hubungannya.

- Generalizations, adalah menggambarkan hubungan khusus dalam obyek anak/child yang menggantikan obyek parent / induk. Dalam hal ini, obyek anak memberikan pengaruhnya dalam hal struktur dan tingkah lakunya kepada obyek induk. Digambarkan dengan garis panah.
- Realizations, merupakan hubungan semantik antara pengelompokkan yang menjamin adanya ikatan diantaranya. Hubungan ini dapat diwujudkan diantara interface dan kelas atau elements, serta antara use cases dan collaborations. Model dari sebuah hubungan realization.

2) Bagan / *Diagram*

UML sendiri terdiri atas pengelompokkan diagram-diagram sistem menurut aspek atau sudut pandang tertentu. Diagram adalah yang menggambarkan permasalahan maupun solusi dari permasalahan suatu model. UML mempunyai 9 diagram, yaitu; use-case, class, object, state, sequence, collaboration, activity, component, dan deployment diagram.

- *Use Case Diagram*, menggambarkan sekelompok use cases dan aktor yang disertai dengan hubungan diantaranya. Diagram use cases ini menjelaskan dan menerangkan kebutuhan / requirement yang diinginkan/ dikehendaki user/pengguna, serta sangat berguna dalam menentukan struktur organisasi dan model dari pada sebuah sistem.
- *Class Diagram***,** yang memperlihatkan struktur statis dari kelas actual didalam sistem.
- *Object* Diagram**,** yang merupakan varian dari kelas diagram yang memperlihatkan lebih detail banyaknya obyek yang mengintantiasi (instances) kelas.
- *State Diagram***,** yang memperliatkan semua keadaan (state) yang dapat dimiliki oleh kelas dan event yang dapat merubah keadaan tersebut.
- *Sequence Diagram***,** yang memperlihatkan kolaborasi dinamik antara objek-objek dengan suatu urutan pesan (a sequence of message) antar objek tersebut.

- *Collaboration Diagram***,** yang memperlihatkan kolaborasi dinamik antar objek tanpa memperhatikan aspek waktu.
- *Activity Diagram*, yang memperlihatkan aliran urutan aktifitas.
- *Component Diagram*, yang memperlihatkan struktur fisik dari source code dalam terminology code components. Komponen berisi informasi tentang logical class dapat berupa komponen source code, komponen biner atau komponen yang dapat dieksekusi.
- *Deployment Diagram*, yang memperlihatkan arsitektur fisik dari hardware dan software pada sistem.