BAB II

TINJAUAN PUSTAKA

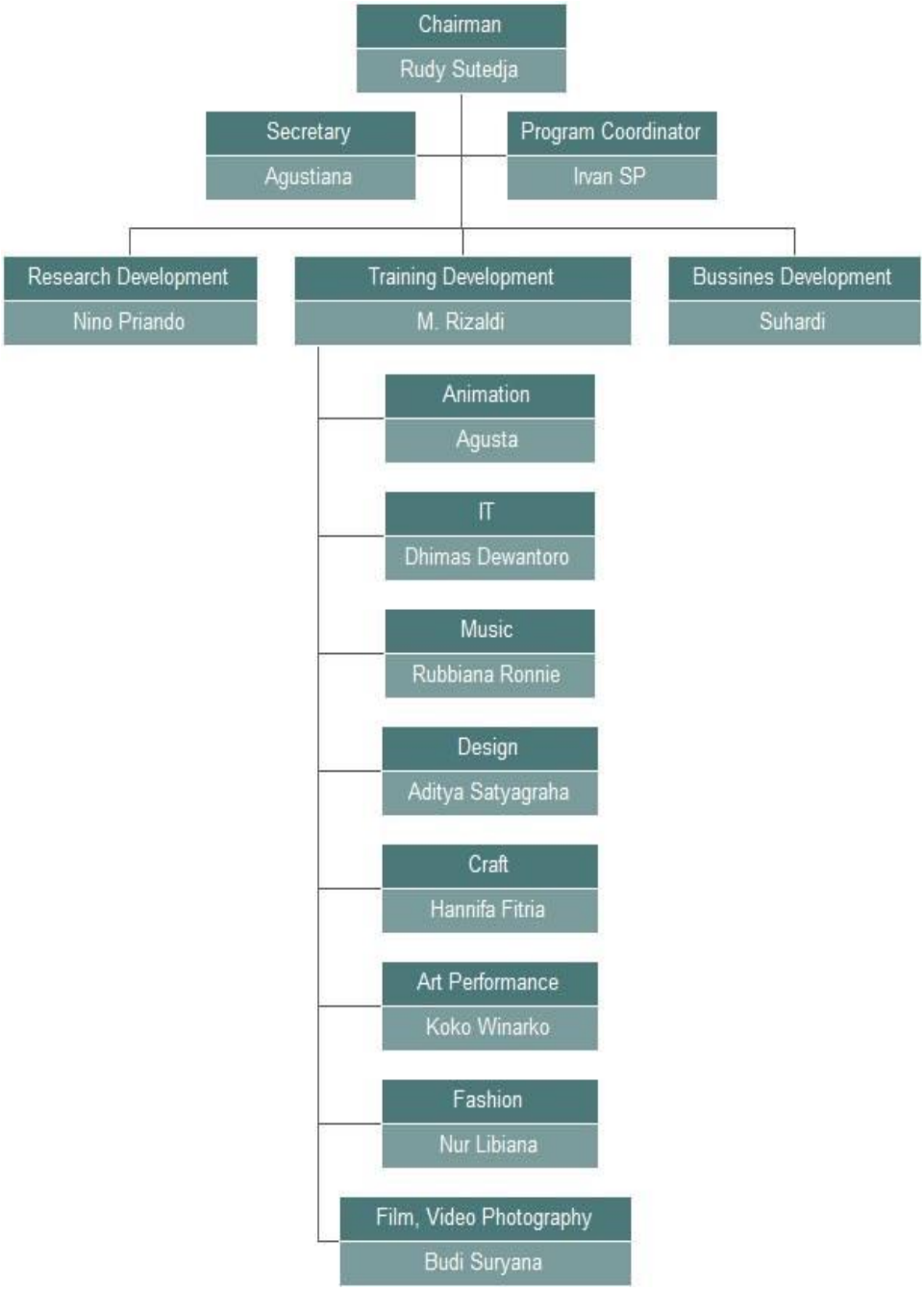
2.1 Profil Tempat Kerja Praktik

2.1.1 Sejarah CCA (Cimahi Creative Association)

Sejak Cimahi menjadi pemerintahan kota pada tahun 2001, telah dirasakan bahwa industry yang berkembang sudah tidak mendapatkan lagi dukungan sumber daya alam sehaingga pada tahun 2004 pemerintah Kota Cimahi melakukan riset bekerja sama dengan Institut Teknologi Bandung (ITB), yang merekomendasikan bahwa Kota Cimahi harus mengembangkan Teknologi Informasi sehingga pada tahun 2005 dideklarasikan "Cimahi Cyber City".

Namun pada perkembangannya pada tahun 2009 hasil riset yang dilakukan oleh "PT. Sembilan Matahari" merekomendasikan bahwa Kota Cimahi harus mengembangkan Animasi dan Film. Sejalan dengan perkembangan didunia dengan fenomena Industri Kreatif, maka pemerintah kota mendeklarasikan untuk mengembangkan Industri Kreatif dengan unggulan IT, Animasi, dan Film. Hal tersebut didukung oleh sumber daya manusia dan potensi pasar yang besar. Maka untuk mendukung rencana tersebut pemerintah kota melaksanakan FGD Industri Kreatif yang pertama yang tujuannya menghimpun pelaku kreatif. Ternyata hasilnya cukup menjanjikan karena jumlah pelaku kreatif cukup signifikan dan memiliki kemampuan yang baik. Pada FGD ke-2 yang dilaksanakan pada tanggal 29 September 2009 maka dibentuk lah Cimahi Creative Association (CCA) berikut pemilihan ketuanya. Pada FGD ke-3, tanggal 28 Oktober 2009, kelembagaan dan kepengurusan CCA diresmikan dengan SK Walikota No. 530/KEP 304 – PENMO/2009. Sedangkan pada FGD ke-4 CCA menyusun Anggaran Dasar (AD), Anggaran Rumah Tangga (ART), serta rencana kerja tahun berikutnya.

2.1.2 Struktur Organisasi CCA (CIMAHI CREATIVE ASSOCIATION)



Gambar 2.1 Struktur Organisasi

2.1.3 Logo CIMAHI CREATIVE ASSOCIATION

Berikut gambar logo CIMAHI CREATIVE ASSOCIATION :



Gambar 2.2 Logo CCA

2.1 Landasan Teori

Landasan teori merupakan penjelasan berbagai konsep dasar dan teori-teori yang berkaitan dalam pembangunan aplikasi *Twitter berbasis web*. Beberapa teori yang terkait dengan pembangunan aplikasi Twitter API, JSON, UML, MYSQL, PHP.

2.2.1 Pengenalan MySQL

MySQL adalah sebuah implementasi dari sistem manajemen basisdata relasional (RDBMS) yang didistribusikan secara gratis dibawah lisensi GPL (General Public License). Setiap pengguna dapat secara bebas menggunakan MySQL, namun dengan batasan perangkat lunak tersebut tidak boleh dijadikan produk turunan yang bersifat komersial. MySQL sebenarnya merupakan turunan salah satu konsep utama dalam basisdata yang telah ada sebelumnya; SQL(Structured Query Language). SQL adalah sebuah konsep pengoperasian basisdata, terutama untuk pemilihan atau seleksi dan pemasukan data, yang memungkinkan pengoperasian data dikerjakan dengan mudah secara otomatis.

Kehandalan suatu sistem basisdata (DBMS) dapat diketahui dari cara kerja pengoptimasi-nya dalam melakukan proses perintah-perintah SQL yang dibuat oleh pengguna maupun program-program aplikasi yang memanfaatkannya. Sebagai peladen basis data, MySQL mendukung operasi basisdata transaksional maupun

operasi basisdata non-transaksional. Pada modus operasi non-transaksional, MySQL dapat dikatakan unggul dalam hal unjuk kerja dibandingkan perangkat lunak peladen basisdata kompetitor lainnya. Namun demikian pada modus non-transaksional tidak ada jaminan atas reliabilitas terhadap data yang tersimpan, karenanya modus non-transaksional hanya cocok untuk jenis aplikasi yang tidak membutuhkan reliabilitas data seperti aplikasi blogging berbasis web (wordpress), CMS, dan sejenisnya. Untuk kebutuhan sistem yang ditujukan untuk bisnis sangat disarankan untuk menggunakan modus basisdata transaksional, hanya saja sebagai konsekuensinya unjuk kerja MySQL pada modus transaksional tidak secepat unjuk kerja pada modus non-transaksional.

- **Keistimewaan MySQL**

Sebagai server database dengan konsep database modern, MySQL memiliki keistimewaan. Beberapa keistimewaan dimiliki MySQL sebagai berikut :

1. ***Portability Database***. MySQL berfungsi dengan stabil tanpa kendala, berarti berlaku pada berbagai sistem operasi seperti Windows, Linux, FreeBSD, Mac OS X Server, Solaris, Amiga, HP-Unix, dan lain-lain.
2. ***Open Source***. MySQL merupakan database open source (gratis), di bawah lisensi GPL sehingga dapat memperoleh dan menggunakannya secara cuma-cuma tanpa membayar sepersen pun.
3. ***Multiuser***. MySQL merupakan database yang dapat digunakan untuk menangani beberapa user dalam waktu bersamaan tanpa mengalami masalah. Dan memungkinkan sebuah database server MySQL dapat diakses client secara bersamaan pula.
4. ***Performace Tuning*.** MySQL mempunyai kecepatan yang cukup baik dalam menangani query query sederhana, serta mampu memproses lebih banyak SQL per satuan waktu.
5. ***Column Type Database***. MySQL didukung dengan tipe data yang sangat kompleks, seperti signed/unsigned integer, float, double, char,

varchar, text, blob, data, time, datetime, timestamp, year, set serta enum.

6. ***Command dan Functions***. MySQL server memiliki operator dan fungsi secara penuh yang mendukung perintah SELECT dan WHERE dalam query.
7. ***Security***. Sistem Security pada MySQL mempunyai beberapa lapisan sekuritas seperti tingkatan subnetmask, hostname, dan izin akses user dengan sistem perizinan yang mendetil serta password terenkripsi.
8. ***Scalability dan Limits***. MySQL mempunyai kemampuan menangani database dalam skala cukup besar, dengan jumlah record lebih dari 50 juta dan 60 ribu tabel serta 5 miliar baris. Selain itu dapat menampung indeks sampai 32 indeks pada tiap tabelnya.
9. ***Connectivity*.** Adanya kemampuan MySQL melakukan koneksi dengan client menggunakan protokol TCP/IP, Unix socket (Unix), atau Named Pipes (NT).
10. ***Localization*.** Adanya kemampuan dalam mendeteksi kesalahan (error code) pada client menggunakan lebih dari dua puluh bahasa.
11. ***Interface*.** MySQL memiliki interface terhadap berbagai aplikasi dan bahasa pemograman menggunakan fungsi API (Application Programming Interface).
12. ***Clients dan Tools***. Database MySQL dilengkapi berbagai tools yang dapat digunakan untuk administrasi database.
13. ***Struktur Tabel*.** MySQL memiliki struktur tabel cukup baik serta cukup fleksibel, misalnya ketika menangani alter table.

2.2.2 JSON

JSON (*JavaScript Object Notation*) adalah format pertukaran data yang ringan, mudah dibaca dan ditulis oleh manusia, serta mudah diterjemahkan dan dibuat (*generate*) oleh komputer. Format ini dibuat berdasarkan bagian dari Bahasa Pemprograman JavaScript, Standar ECMA-262 Edisi ke-3 – Desember 1999. JSON

merupakan format teks yang tidak bergantung pada bahasa 45 pemrograman apapun karena menggunakan gaya bahasa yang umum digunakan oleh *programmer* keluarga C termasuk C, C++, C#, Java, JavaScript, Perl, Python dll. Oleh karena sifat-sifat tersebut, menjadikan JSON ideal sebagai bahasa pertukaran-data.

JSON terbuat dari dua struktur:

1. Kumpulan pasangan nama/nilai. Pada beberapa bahasa, hal ini dinyatakan sebagai objek (object), rekaman (record), struktur (struct), kamus (dictionary), tabel hash (hash table), daftar berkunci (keyed list), atau associative array.

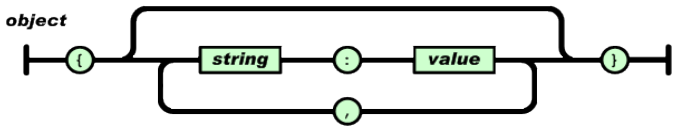
2. Daftar nilai terurutkan (an ordered list of values). Pada kebanyakan bahasa, hal ini dinyatakan sebagai larik (array), vektor (vector), daftar (list), atau urutan (sequence).

Struktur-struktur data ini disebut sebagai struktur data universal. Pada dasarnya, semua bahasa pemprograman moderen mendukung struktur data ini dalam bentuk yang sama maupun berlainan. Hal ini pantas disebut demikian karena format data mudah dipertukarkan dengan bahasa-bahasa pemprograman yang juga berdasarkan pada struktur data ini.

JSON menggunakan bentuk sebagai berikut:

1. Objek

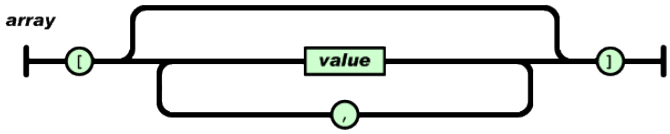
Objek adalah sepasang nama/nilai yang tidak terurutkan. Objek dimulai dengan { (kurung kurawal buka) dan diakhiri dengan } (kurung kurawal tutup). Setiap nama diikuti dengan : (titik dua) dan setiap pasangan nama/nilai dipisahkan oleh , (koma).



Gambar 2.3 Objek JSON

2. Array

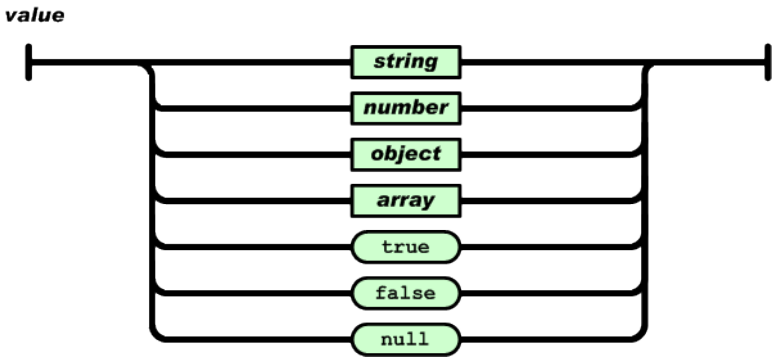
Array adalah kumpulan nilai yang terurutkan. Array dimulai dengan [(kurung kotak buka) dan diakhiri dengan] (kurung kotak tutup). Setiap nilai dipisahkan oleh , (koma).



Gambar 2.4 Array JSON

3. Nilai (Value)

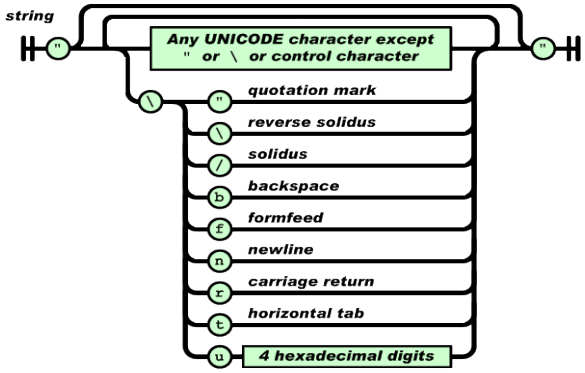
Nilai (*value*) dapat berupa sebuah string dalam tanda kutip ganda, atau angka, atau *true* atau *false* atau null, atau sebuah objek atau sebuah larik. Struktur-struktur tersebut dapat disusun bertingkat.



Gambar 2.5 Nilai JSON 47

4. String

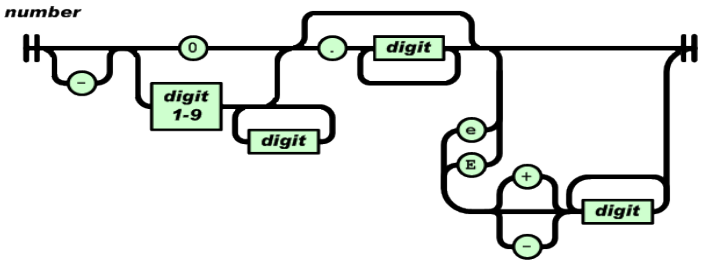
String adalah kumpulan dari nol atau lebih karakter *Unicode*, yang dibungkus dengan tanda kutip ganda. Di dalam string dapat digunakan *backslash escapes* "\" untuk membentuk karakter khusus. Sebuah karakter mewakili karakter tunggal pada string. String sangat mirip dengan string C atau Java.



Gambar 2.6 String JSON

5. Number (Angka)

Angka adalah sangat mirip dengan angka di C atau Java, kecuali format oktal dan heksadesimal tidak digunakan.



Gambar 2.7 Angka JSON 48

2.2.3 *Tools* Implementasi

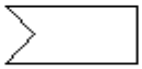
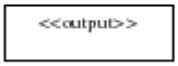
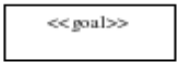
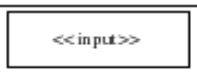
Tools implementasi merupakan peralatan-peralatan yang dibutuhkan dalam membangun perangkat lunak. Adapun *tools* implementasi yang digunakan untuk

membuat aplikasi *Twitter* berbasis Web adalah Dreamweaver CS5 sebagai perangkat pembangun PHP, dan MySQL sebagai sistem manajemen basis data.

2.3 UML (Unified Modeling Language)

UML Adalah sebuah bahasa untuk menentukan, visualisasi, kontruksi dan mendokumentasikan artifact (bagian dari informasi yang digunakan atau dihasilkan dalam suatu proses pembuatan perangkat lunak. Artifact dapat berupa model, deskripsi atau perangkat lunak) dari system perangkat lunak, seperti pada pemodelan bisnis dan system non perangkat lunak. UML merupakan suatu kumpulan teknik terbaik yang telah terbukti sukses dalam memodelkan system yang besar dan kompleks.

Table 1 Symbol – symbol UML.

Notasi	Deskripsi
 Actor	Menggambarkan pelaku / user dalam sebuah sistem. Pelaku ini meliputi manusia atau sistem komputer atau sub sistem lain yang memiliki method untuk melakukan sesuatu. *Contoh :* Customer, Manager, Application Server, dan lain-lain
 Event	Menggambarkan kejadian *(event)* atau peristiwa yang mendorong terjadinya proses. *Contoh :* Adanya Order, Jatuh Tempo, kebutuhan buku, dan lain-lain
 Process	Kumpulan aktivitas yang dirancang dalam tujuan tertentu untuk menghasilkan output khusus. Dapat juga dikatakan, proses merupakan rangkaian aktivitas dalam waktu dan tempat tertentu, yang memiliki awal dan akhir aktivitas; memiliki input dan output yang jelas. *Contoh :* Mengumpulkan data, Analisis Data, dan lain-lain
 Output	Sebuah proses dapat menghasilkan satu atau lebih output yang memenuhi kebutuhan eksternal atau sesuai dengan penggunaan internal. Output dapat berupa obyek fisik (seperti laporan *atau* invoice), rekomendasi atau hasil keseluruhan dari proses. Dalam beberapa kasus, output ini dapat menjadi sebuah *event* yang dapat mendorong terjadinya proses baru.
 Goal	Tujuan yang ingin dicapai dalam sebuah proses.
	Input, Resources dan Information yang digunakan dalam process.

Bagian-Bagian Uml

Bagian-bagian utama dari UML adalah view, diagram, model element, dan general mechanism.

a. **View**

View digunakan untuk melihat sistem yang dimodelkan dari beberapa aspek yang berbeda. View bukan melihat grafik, tapi merupakan suatu abstraksi yang berisi sejumlah diagram.

Beberapa jenis view dalam UML antara lain: use case view, logical view, component view, concurrency view,dan deployment view.

b. Use case view

Mendeskripsikan fungsionalitas sistem yang seharusnya dilakukan sesuai yang diinginkan external actors. Actor yang berinteraksi dengan sistem dapat berupa user atau sistem lainnya.

View ini digambarkan dalam use case diagramsdan kadang-kadang dengan activity diagrams. Viewini digunakan terutama untuk pelanggan, perancang (designer), pengembang (developer), dan penguji sistem (tester).

c. Logical view

Mendeskripsikan bagaimana fungsionalitas dari sistem, struktur statis (class, object,danrelationship) dan kolaborasi dinamis yang terjadi ketika object mengirim pesan ke object lain dalam suatu fungsi tertentu.

View ini digambarkan dalam class diagrams untuk struktur statis dan dalam state, sequence, collaboration, dan activity diagram untuk model dinamisnya. View ini digunakan untuk perancang (designer) dan pengembang (developer).

d. Component view

Mendeskripsikan implementasi dan ketergantungan modul. Komponen yang merupakan tipe lainnya dari code module diperlihatkan dengan struktur dan ketergantungannya juga alokasi sumber daya komponen dan informasi administrative lainnya.

View ini digambarkan dalam component view dan digunakan untuk pengembang (developer).

e. Concurrency view

Membagi sistem ke dalam proses dan prosesor.View ini digambarkan dalam diagram dinamis (state, sequence, collaboration, dan activity diagrams) dan diagram implementasi (component dan deployment diagrams) serta digunakan untuk pengembang (developer), pengintegrasi (integrator), dan penguji (tester).

f. Deployment view

Mendeskripsikan fisik dari sistem seperti komputer dan perangkat (nodes) dan bagaimana hubungannya dengan lainnya.

View ini digambarkan dalam deployment diagramsdan digunakan untuk pengembang (developer), pengintegrasi (integrator), dan penguji (tester).

g. Diagram

Diagram berbentuk grafik yang menunjukkan simbol elemen model yang disusun untuk mengilustrasikan bagian atau aspek tertentu dari sistem. Sebuah diagram merupakan bagian dari suatu view tertentu dan ketika digambarkan biasanya dialokasikan untuk view tertentu. Adapun jenis diagram antara lain :

1. Use Case Diagram

Use case adalah abstraksi dari interaksi antara system dan actor. Use case bekerja dengan cara mendeskripsikan tipe interaksi antara user sebuah system dengan sistemnya sendiri melalui sebuah cerita bagaimana sebuah system dipakai. Use casemerupakan konstruksi untuk mendeskripsikan bagaimana system akan terlihat di mata user. Sedangkan use case diagram memfasilitasi komunikasi diantara analis dan pengguna serta antara analis dan client.

2. Class Diagram

Class adalah dekripsi kelompok obyek-obyek dengan property, perilaku (operasi) dan relasi yang sama. Sehingga dengan adanya class diagram dapat memberikan pandangan global atas sebuah system. Hal tersebut tercermin dari class- class yang ada dan relasinya satu dengan yang lainnya. Sebuah sistem biasanya mempunyai beberapa class diagram. Class diagram sangat membantu dalam visualisasi struktur kelas dari suatu system.

3. Component Diagram

Component software merupakan bagian fisik dari sebuah system, karena menetap di komputer tidak berada di benak para analis. Komponent merupakan implementasi software dari sebuah atau lebih class. Komponent dapat berupa source code, komponent biner, atau executable component. Sebuah komponent berisi informasi tentang logic class atau class yang diimplementasikan sehingga membuat pemetaan dari logical view ke component view.Sehingga component diagram merepresentasikan dunia riil yaitu component software yang mengandung component, interface dan relationship.

4. Deployment Diagram

Menggambarkan tata letak sebuah system secara fisik, menampakkan bagian-bagian software yang berjalan pada bagian-bagian hardware, menunjukkan hubungan komputer dengan perangkat (nodes) satu sama lain dan jenis hubungannya. Di dalam nodes,executeable component dan object yang

dialokasikan untuk memperlihatkan unit perangkat lunak yang dieksekusi oleh node tertentu dan ketergantungan komponen.

5. State Diagram

Menggambarkan semua state (kondisi) yang dimiliki oleh suatu object dari suatu class dan keadaan yang menyebabkan state berubah. Kejadian dapat berupa object lain yang mengirim pesan. State class tidak digambarkan untuk semua class, hanya yang mempunyai sejumlah state yang terdefinisi dengan baik dan kondisi class berubah oleh stateyang berbeda.

6. Sequence Diagram

Sequence Diagram digunakan untuk menggambarkan perilaku pada sebuah scenario. Kegunaannya untuk menunjukkan rangkaian pesan yang dikirim antara object juga interaksi antaraobject, sesuatu yang terjadi pada titik tertentu dalam eksekusi sistem.

7. Collaboration Diagram

Menggambarkan kolaborasi dinamis sepertisequence diagrams. Dalam menunjukkan pertukaran pesan, collaboration diagrams menggambarkan objectdan hubungannya (mengacu ke konteks).Jika penekannya pada waktu atau urutan gunakansequencediagrams, tapi jika penekanannya pada konteks gunakan collaboration diagram.

8. Activity Diagram

Menggambarkan rangkaian aliran dari aktivitas, digunakan untuk mendeskripsikan aktifitas yang dibentuk dalam suatu operasi sehingga dapat juga digunakan untuk aktifitas lainnya seperti use caseatau interaksi.

2.4 API (*Application Programming Interface*)

API adalah sekumpulan perintah, fungsi, dan protokol yang dapat digunakan saat membangun perangkat lunak untuk sistem operasi tertentu. API memungkinkan programmer untuk menggunakan fungsi standar untuk berinteraksi dengan sistem operasi. API atau Application Programming Interface juga merupakan suatu dokumentasi yang terdiri dari antar muka, fungsi, kelas, struktur untuk membangun sebuah perangkat lunak. Dengan adanya API, maka memudahkan seorang programmer untuk membongkar suatu software untuk kemudian dapat dikembangkan atau diintegrasikan dengan perangkat lunak yang lain. API dapat dikatakan sebagai penghubung suatu aplikasi dengan aplikasi lainnya. Suatu rutin standar yang memungkinkan developer menggunakan system function. Proses ini dikelola melalui operating system. Keunggulan dari API ini adalah memungkinkan suatu aplikasi dengan aplikasi lainnya untuk saling berinteraksi.

Keuntungan dengan menggunakan API adalah sebagaiberikut:

1) Portabilitas.

   Developer yang menggunakan API dapat menjalankan programnya dalam sistem operasi mana saja asalkan sudah ter- *install* API tersebut.

2) Lebih Mudah Dimengerti

   API menggunakan bahasa yang lebih terstruktur dan mudah dimengerti daripada bahasa *system call*. Hal ini sangat penting dalam hal *editing* dan pengembangan.

System call interface ini berfungsi sebagai penghubung antara API dan *system call* yang dimengerti oleh sistem operasi. *System call interface* ini akan menerjemahkan perintah dalam API dan kemudian akan memanggil *system calls* yang diperlukan. Untuk membuka suatu *file* tersebut *user* menggunakan program yang telah dibuat dengan menggunakan bantuan API, maka perintah dari *user* tersebut diterjemahkan dulu oleh program menjadi perintah *open()*. Perintah *open()* ini merupakan perintah dari API dan

bukan perintah yang langsung dimengerti oleh kernel sistem operasi. Oleh karena itu, agar keinginan pengguna dapat dimengerti oleh sistem operasi, maka perintah *open()* tadi diterjemahkan ke dalam bentuk *system call* oleh *system call interface*. Implementasi perintah *open()* tadi bisa bermacam-macam tergantung dari sistem operasi yang digunakan.

Cara menggunakan API :

1. Dilakukan dengan mengimpor package/kelas
2. Ada beberapa kelas bernama sama dipackage yang berbeda, yaitu :
    - import salah satu dan gunakan nama lengkap untuk yang lain
    - gunakan nama lengkap semua kelas

Kebanyakan Sistem Operasi seperti Windows, menyediakan fasilitas API sehingga programmer dapat melakukan aktivitas programming dengan lebih konsisten. Meskipun API didesain untuk programer, namun API juga baik untuk *user* karena setidaknya dapat menjamin bahwa program tersebut memiliki *interface* yang sama, sehingga lebih mudah untuk dipelajari

2.5 Img.ly

Img.ly adalah salah satu layanan internet yang menyediakan jasa penyimpanan gambar (*Image hosting)* . Sebuah jasa penyimpanan gambar memungkinkan pengguna untuk mengunggah gambar ke sebuah website internet. Gambar tersebut lalu disimpan di server penyedia layanan dan selanjutnya bisa dilihat atau digunakan untuk keperluan yang beragam. Gambar 2.5 adalah logo yang digunakan oleh img.ly.



Gambar 2.8 Logo img.ly

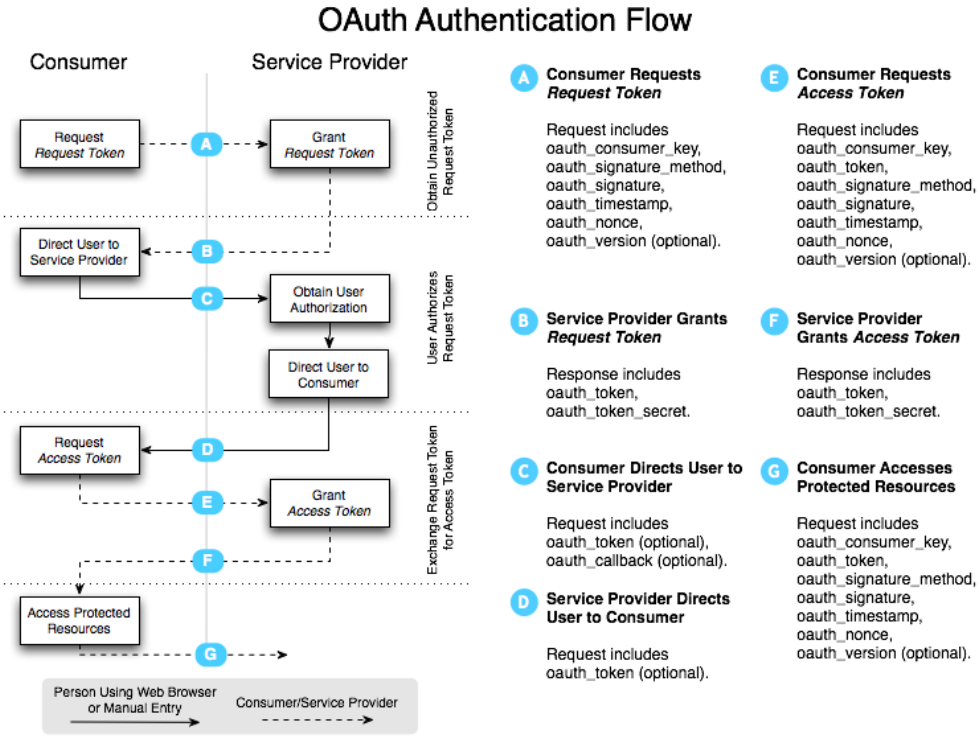
2.6 Oauth

Oauth adalah suatu protokol terbuka yang memungkinkan pengguna untuk berbagi sumber pribadi mereka (mis. foto, video, daftar alamat) yang disimpan di suatu situs web dengan situs lain tanpa perlu menyerahkan nama pengguna dan kata sandi mereka. Proses ini dilakukan dengan memberikan *token*, bukan nama pengguna dan kata sandi, untuk data mereka yang diinangi oleh suatu penyedia jasa tertentu. Setiap *token* memberikan akses untuk suatu situs spesifik terhadap suatu sumber spesifik selama durasi tertentu. Gambar 2.7 adalah merupakan logo dari Oauth.



Gambar 2.9 Logo Oauth

OAuth mengizinkan seorang mengguna untuk memberikan akses kepada situs pihak ketiga untuk mengakses informasi mereka yang disimpan di penyedia layanan lain tanpa harus membagi izin akses atau keseluruhan data mereka
Gambar 2.8 merupakan gambaran cara kerja *Oauth*.



Gambar 2.10 OAuth Authentication Flow

2.6.1 Cara Kerja *Oauth* di Twitter

Ketika pengunjung datang ke halaman *login* , aplikasi mengirimkan request GET ke server *twitter* melalui sebuah set kode *request.* Kode *request* ini digunakan untuk verifikasi pengguna pada website *twitter.*

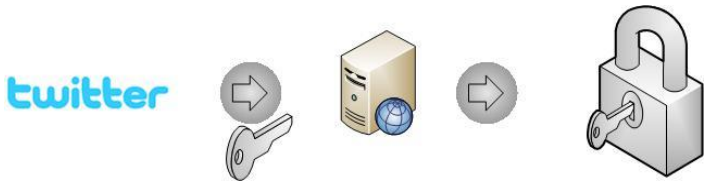
Pengguna lalu di dibawa ke halaman *twitter* untuk melakukan *allow* atau *deny* untuk akses aplikasi. Jika pengguna melakukan *allow* pada aplikasi , maka pengguna akan dibawa kembali ke alamat aplikasi. URL tempat pengguna akan dipindahkan telah berisi *token OAuth* yang akan digunakan di langkah selanjutnya. Urutan cara kerja *OAuth* pada *twitter* adalah sebagai berikut:

1. Pengguna melakukan akses pada aplikasi , aplikasi menghubungi *twitter* untuk mendapatkan *one-time Request Token* seperti pada gambar 2.9



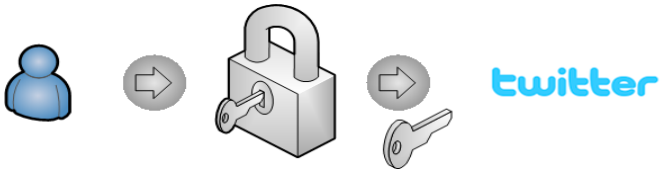
Gambar 2.11 Cara kerja OAuth-01

2. *Twitter* membalas dengan *Request Token* dan *Request Token Secret Key*.*Request Token* digunakan untuk membuat *authorization lin*k, dan *Request Token Secret* digunakan untuk memberitahu twitter bahwa user adalah yang asli melakukan *request* untuk *Request Token* seperti pada gambar 2.10



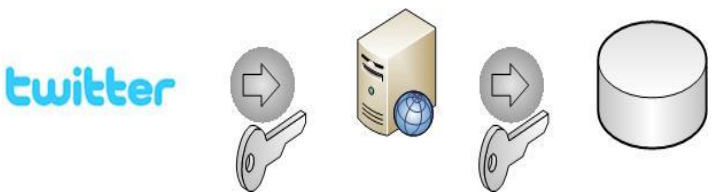
Gambar 2.12 Cara Kerja Oauth-02

3. User melakukan klik pada *authorization link* yang baru dibangkitkan, dibuat dengan *Request Token. Link* melakukan redirect ke *Twitter*, dan *API call* melakukan pass *Request Token Secret* seperti pada gambar 2.11



Gambar 2.13 Cara Kerja OAuth-03

4. *Twitter* memberi prompt kepada user untuk melakukan *allow* untuk akses aplikasi. Jika pengguna melakukan konfirmasi , *Twitter* mengirimkan pengguna kembali ke alamat aplikasi dengan *Access Token* dan *Access Token Secret key* . *Token key/pair* adalah *unique* untuk setiap pengguna dan tidak berubah. *Token key/pair* ini bisa disimpan di database untuk keperluan mendatang seperti pada gambar 2.12



Gambar 2.14 Cara Kerja Oauth-04

2.6.2 Mendaftarkan aplikasi di Twitter

Untuk memulai menggunakan API *twitter* , diharuskan untuk mendaftarkan terlebih dahulu dengan langkah sebagai berikut :

1. Melakukan akses ke http://dev.twitter.com/apps/new

2. Mengisi hal-hal yang diperlukan yaitu :

   a. Application name : nama aplikasi
   b. Description : deskripsi aplikasi
   c. Application Website : alamat website dari aplikasi
   d. Organization : organisasi pemilik aplikasi
   e. Application type : tipe aplikasi yaitu Client atau Browser
   f. Callback URL : alamat URL untuk di *redirect* setelah akses disetujui
   g. Default Access Type : berupa *"Read & Write"* atau *"Read Only"*
   h. Application Icon : Icon untuk aplikasi
   i. Captcha : form yang harus diisi untuk memastikan bahwa yang membuat adalah manusia

3. Setelah pembuatan/pendaftara aplikasi selesai maka akan diberikan *Consumer key , Consumer secret , API key , Access Token* dan Access *Token Secret.*

2.7 PHP

PHP adalah bahasa pemrograman *script* yang paling banyak dipakai saat ini. PHP banyak dipakai untuk memrogram situs web dinamis, walaupun tidak tertutup kemungkinan digunakan untuk pemakaian lain.

Contoh terkenal dari aplikasi PHP adalah phpBB dan MediaWiki (software di belakang Wikipedia). PHP juga dapat dilihat sebagai pilihan lain dari ASP.NET/C#/VB.NET Microsoft, ColdFusion Macromedia, JSP/Java Sun Microsystems , dan CGI/Perl. Contoh aplikasi lain yang lebih kompleks berupa CMS yang dibangun menggunakan PHP adalah Mambo, Joomla!, Postnuke, Xaraya, dan lain-lain.

2.7.1 Sejarah PHP

Pada awalnya PHP merupakan kependekan dari *Personal Home Page* (Situs Personal). PHP pertama kali dibuat oleh Rasmus Lerdorf pada tahun 1995. Pada waktu itu PHP masih bernama FI (*Form Interpreted*), yang wujudnya berupa sekumpulan *script* yang digunakan untuk mengolah data *form* dari web. Logo dari PHP adalah seperti gambar 2.13 .



Gambar 2.15 Logo PHP

Selanjutnya Rasmus merilis kode sumber tersebut untuk umum dan menamakannya PHP/FI. Dengan perilisan kode sumber ini menjadi *open source*, maka banyak programmer yang tertarik untuk ikut mengembangkan PHP.

Pada November 1997, dirilis PHP/FI 2.0. Pada rilis ini interpreter PHP sudah diimplementasikan dalam program C. Dalam rilis ini disertakan juga modul-modul ekstensi yang meningkatkan kemampuan PHP/FI secara signifikan.

Pada tahun 1997, sebuah perusahaan bernama Zend menulis ulang interpreter PHP menjadi lebih bersih, lebih baik, dan lebih cepat. Kemudian pada Juni 1998, perusahaan tersebut merilis interpreter baru untuk PHP dan

meresmikan rilis tersebut sebagai PHP 3.0 dan singkatan PHP dirubah menjadi akronim berulang *PHP: Hypertext Preprocessing*.

Pada pertengahan tahun 1999, Zend merilis interpreter PHP baru dan rilis tersebut dikenal dengan PHP 4.0. PHP 4.0 adalah versi PHP yang paling banyak dipakai pada awal abad ke-21. Versi ini banyak dipakai disebabkan kemampuannya untuk membangun aplikasi web kompleks tetapi tetap memiliki kecepatan dan stabilitas yang tinggi.

Pada Juni 2004, Zend merilis PHP 5.0. Dalam versi ini, inti dari interpreter PHP mengalami perubahan besar. Versi ini juga memasukkan model pemrograman berorientasi objek ke dalam PHP untuk menjawab perkembangan bahasa pemrograman ke arah paradigma berorientasi objek.

2.7.2 Kelebihan PHP

1. Bahasa pemrograman PHP adalah sebuah bahasa *script* yang tidak melakukan sebuah kompilasi dalam penggunaanya.
2. Web Server yang mendukung PHP dapat ditemukan dimana - mana dari mulai apache, IIS, Lighttpd, nginx, hingga Xitami dengan konfigurasi yang relatif mudah.
3. Dalam sisi pengembangan lebih mudah, karena banyaknya milis - milis dan developer yang siap membantu dalam pengembangan.
4. Dalam sisi pemahamanan, PHP adalah bahasa *scripting* yang paling mudah karena memiliki referensi yang banyak.
5. PHP adalah bahasa open source yang dapat digunakan di berbagai mesin (Linux, Unix, Macintosh, Windows) dan dapat dijalankan secara runtime melalui console serta juga dapat menjalankan perintah-perintah sistem.

2.8 JavaScript

JavaScript adalah bahasa *script* yang digunakan pada halaman HTML. *JavaScript* adalah varian dari *java* yang sangat terkenal dalam lingkungan pemrograman *web*. *JavaScript* mempunyai keunggulan pada sisi *client*, maupun diakses lebih cepat dari *script server*. Tetapi kelemahannya pada *script* ini belum tentu mampu diproses oleh *browser client* tergantung kompetibilitas *browser* tersebut. Cara untuk menjalankan j*avascript* hanya dengan *javascript-enabled* yaitu *browser* yang mampu menjalankan *javascript* seperti *Netscape Navigator* (versi 2.0 keatas) atau *internet ekplorer* (MSIE-versi 3.0 ke atas).

Beberapa hal mengenai *JavaScript* :

1. *JavaScript* didesain untuk menambah interaktif suatu *web*.
2. *JavaScript* merupakan sebuah bahasa *scripting* dengan bahasa pemrograman yang ringan.
3. *JavaScript* biasanya disisipkan (*embedded*) dalam halaman HTML.
4. *JavaScript* adalah bahasa interpreter (yang berarti skrip dieksekusi tanpa proses kompilasi.