

BAB II

LANDASAN TEORI

2.1. *Server*

2.1.1. Pengertian

Server adalah sebuah sistem komputer yang menyediakan jenis layanan (*service*) tertentu dalam sebuah jaringan komputer. *Server* didukung dengan *processor* yang bersifat *scalable* dan RAM (*Random Access Memory*) yang besar, juga dilengkapi dengan sistem operasi khusus, yang disebut sebagai sistem operasi jaringan (*network operating system*). *Server* juga menjalankan perangkat lunak administratif yang mengontrol akses terhadap jaringan dan sumber daya yang terdapat di dalamnya, seperti halnya berkas atau alat pencetak (*printer*), dan memberikan akses kepada *workstation* anggota jaringan.

server dibagi menjadi beberapa jenis sesuai dengan fungsinya, berikut jenis *server* :

1. *Server Aplikasi (Application server)*
2. *Server Data (Data Server)*
3. *Server Proxy (Proxy Server)*.

Tugas utama *server* adalah melayani komputer *client*, dan di bagi menjadi beberapa fungsi sesuai dengan jenis *server*, berikut penjelasannya :

1. *Server Aplikasi*

Server yang digunakan untuk menyimpan berbagai macam aplikasi yang dapat diakses oleh *client*.

2. *Server Data*

Server jenis ini di gunakan untuk menyimpan berbagai data, baik data yang belum diolah ataupun data yang sudah diolah menjadi informasi. Data ini dapat di akses oleh *client* dengan bantuan aplikasi yang ada di *server*.

3. *Server Proxy*

Sedangkan *Server proxy* berfungsi untuk mengatur lalu lintas di jaringan melalui pengaturan *proxy*. Orang awam lebih mengenal *proxy server* untuk mengoneksikan komputer *client* ke *Internet*.

2.1.2. Manfaat *Server*

Dengan menggunakan sebuah komputer *server*, berbagai biaya dan juga waktu dapat dipangkas, sehingga sebuah kegiatan menjadi lebih ekonomis. Misalnya, jika di sebuah perusahaan terdapat sebuah komputer *server* yang terhubung ke semua komputer lain sebagai kliennya, maka sebuah data dapat dibagikan ke sesama klien dalam jaringan perusahaan tersebut.



Gambar 2.1 *Server*

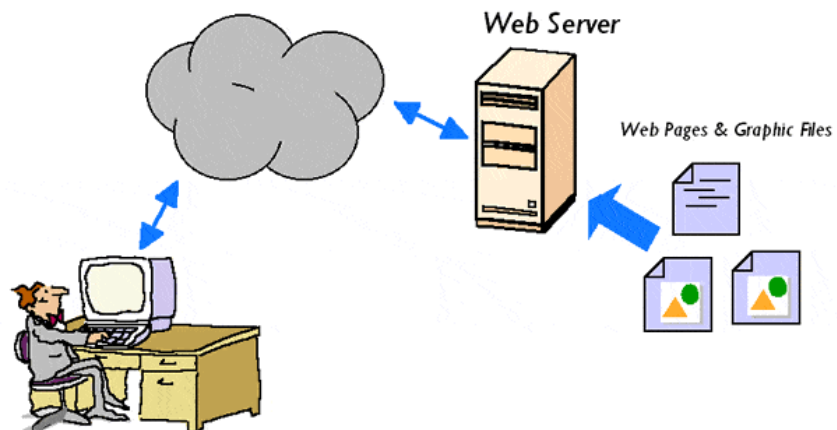
Kemudian jika misalnya ada komputer klien yang ingin mencetak sebuah *file*, maka bisa langsung melalui komputer *server*, sehingga hanya dibutuhkan satu buah *printer* saja. Tentu jauh lebih ekonomis dibandingkan dengan jika harus melakukan pemasangan *printer* untuk tiap komputer yang ada di perusahaan tersebut. Untuk membuat sebuah komputer *server*, terdapat beberapa hal yang harus diperhatikan, seperti :

1. *RAM (Random Access Memory)* : kapasitas memori yang besar agar proses *multitasking* menjadi lebih cepat.
2. *Processor* : komputer *server* sebaiknya memiliki kecepatan akses *processor* yang mumpuni agar kinerja tetap terjaga dan tidak *down*.
3. *Hard Drive* : berguna untuk menyimpan berbagai macam data komputer klien yang terpusat pada komputer *server*. Kebanyakan *server* canggih lebih memilih menggunakan *SSD (Solid State*

Drive/Disk) ketimbang *hard drive* atau *harddisk* karena performa atau kinerjanya lebih baik.

2.1.3. Cara Kerja *Server*

Secara sederhana, *server* bekerja atas permintaan dari sebuah klien. Misalnya, untuk kasus *web server*, ketika Anda mengetikkan suatu alamat *website* menggunakan *browser*, maka artinya komputer Anda sedang bertindak sebagai klien yang meminta informasi kepada *web server*. *Web server* tersebut kemudian mengirimkan isi *website* ke komputer Anda, sehingga Anda pun dapat mengakses isi *website* tersebut.



Gambar 2.2 Cara Kerja *Server*

Untuk kasus lainnya, seperti *server FTP* (*File Transfer Protocol*), mungkin agak sedikit berbeda. Pada *server FTP*, Anda dapat mengunggah sebuah dokumen atau data menuju *server FTP*, sehingga dapat disimpan dalam *server* tersebut. Sebagai klien, Anda berhak untuk menyimpan data Anda di *server FTP*.

Nantinya, jika ada orang lain yang tergabung dalam jaringan *server* tersebut dan ingin mengunduh data atau dokumen Anda, maka *server* FTP akan menyediakan koneksi untuk klien lain tersebut. Secara umum, semua jenis *server* bekerja dengan menjalankan fungsi – fungsi yang telah disebutkan sebelumnya, mulai dari melayani permintaan data dari klien hingga memberikan perlindungan pada komputer klien. Hanya saja, untuk jenis *server* yang berbeda, hal yang dilayani pun berbeda.

2.2. Web Streaming Server

Selain *server* kita juga perlu mengenal *streaming server* untuk melihat *video* yang ada di *server* tersebut. *Streaming Server* adalah sebuah *web server* atau aplikasi yang terinstal di dalam sebuah *server* yang digunakan untuk menjalankan file video atau audio secara *real-time* atau *streaming* di internet. Dengan kata lain, *file video* ataupun *audio* yang terletak dalam sebuah *server* dapat secara langsung dijalankan setelah ada permintaan dari *user*, sehingga proses running aplikasi yang *download* berupa waktu yang lama dapat dihindari tanpa harus melakukan proses penyimpanan terlebih dahulu. Saat *file video* atau *audio* di *stream*, akan berbentuk sebuah *buffer* di komputer *client*, dan data *video - audio* tersebut akan mulai di download ke dalam *buffer* yang telah terbentuk pada mesin *client*. Dalam waktu sepersekian detik, *buffer* telah terisi penuh dan secara otomatis *audio* dijalankan oleh sistem. Sistem akan membaca informasi dari *buffer* dan

tetap melakukan proses *download file*, sehingga proses *streaming* tetap berlangsung.

Streaming server mengizinkan kita untuk meletakkan *file-file audio* atau *video* secara terpisah dari *web server* yang kita jalankan. Situs-situs yang menyediakan layanan *streaming video* atau *audio* menggunakan *streaming server* untuk menjalankan layanannya. Contoh situs-situs yang menggunakan *streaming server* di antaranya *Youtube*, *Metacafe*, dan *Megavideo*.

Media server menangani pengolahan aset digital yang diolah sedemikian rupayang kemudian didistribusi kepada klien. *Media server* bisa diakses oleh semua perangkat yang berada pada jaringan yang terhubung dengan *media server* tersebut. *Media server* bertugas sebagai media yang mengolah aset digital yang menggunakan NAS (*Network Attached Storage*) sebagai media penyimpanan dan *web server* sebagai *interface* yang digunakan untuk bertinteraksi dengan user.

Sebuah *server media* bisa menunjuk pada alat komputer khusus atau perangkat lunak aplikasi khusus, mulai dari mesin kelas enterprise yang menyediakan *video on demand* , untuk lebih umum, sebuah komputer kecil pribadi atau NAS (*Network Attached Storage*) untuk rumah, khususnya untuk menyimpan berbagai mediadigital (seperti *video digital/film*, *audio /musik*, dan *file gambar*). Satu – satunya yang diperlukan untuk server media adalah metode penyimpanan media dan koneksi jaringan dengan bandwidth yang cukup untuk memungkinkan akses ke media. Tergantung

pada penggunaan dan aplikasi yang dijalankan, media server mungkin memerlukan sejumlah besar *RAM*, atau multicore *CPU* yang kuat.

Meningkatnya penggunaan grafis gerak dalam lingkungan seperti Teater, Tari, Acara Perusahaan dan wisata *rock* telah menyebabkan perkembangan *server media* yang dirancang khusus untuk acara live. Dalam dunia telepon, *server media* adalah komponen komputasi yang memproses *audio* dan *video stream* yang berhubungan dengan panggilan telepon atau koneksi. Layanan konferensi adalah contoh bagaimana *server media* dapat digunakan. Dengan jaringan telepon bergerak lebih ke arah teknologi *VoIP*.

2.3. *Vimp*

ViMP adalah *CMS Video* profesional untuk *Web TV*, Portal Social Media dan Video Komunitas. *ViMP* dirancang untuk portal dengan trafik tinggi, video dengan resolusi tinggi dan platform *Web TV* dan video komunitas dengan lingkungan yang kuat. *ViMP* memberikan panel administratif yang lengkap dan memungkinkan perubahan yang mudah dan cepat dan merealisasikan kebutuhan grafik kita. Dapat memilih dari tiga layout yang ada dan mengintegrasikan logo sesuai keinginan. Bahkan jika ingin dapat membuat layout sendiri untuk *ViMP*. *ViMP flash player* juga dapat diintegrasikan ke halaman web lainnya dengan usaha yang sangat kecil, karena pembuatan embed code yang otomatis. Oleh karena itu kita dapat membuka akses portal kita ke yang lain dengan mudah.



Gambar 2.3 *Vimp*

2.4. *Ubuntu Server*

Ubuntu Server adalah sistem operasi *server*, yang dikembangkan oleh *Canonical*, yang berjalan pada semua arsitektur utama: x86, x86-64, ARM v7, ARM64, POWER8, dan *IBM System z mainframe via LinuxONE*. *Ubuntu* adalah *platform server* yang dapat digunakan siapapun untuk hal berikut dan banyak lagi :

1. *Situs web*
2. *FTP*
3. *Server email*
4. *File dan server cetak*
5. *Platform pengembangan*
6. *Penyebaran kontainer*
7. *Layanan awan*
8. *Server database*

Salah satu kegunaan yang membuat *Ubuntu Server* begitu menarik adalah hemat biaya. Siapa pun dapat mendownload salinan versi terbaru dari *Ubuntu Server* dan menyebarkannya sebanyak mungkin mesin – dengan biaya nol (dikurangi perangkat keras dan waktu).

2.4.1. *Editor Nano*

Editor nano dirancang untuk meniru fungsionalitas dan kemudahan penggunaan *PICO (Pine Composer) text editor*. Ada empat bagian utama *editor*. Bagian atas baris menunjukkan versi program, nama *file* saat ini yang sedang diedit, dan apakah atau tidak ada *file* yang telah dimodifikasi. Selanjutnya adalah jendela *editor* utama yang menunjukkan *file* yang sedang diedit. Baris status adalah baris ketiga dari bawah dan menunjukkan pesan penting. Kedua bawah garis menunjukkan yang paling umum digunakan. Menjalankan *editor nano* adalah sebagai berikut :

Nano nama *file* : membuat *file* teks

Tabel 2.1 Daftar Perintah *Editor Nano*

^G	<i>Help</i>	^W	Mencari kata
^O	Menyimpan <i>file</i>	^U	<i>Undo</i> untuk perintah menghapus
^R	Membuka <i>file</i>	^C	Menampilkan posisi <i>cursor</i> saat ini
^K	Menghapus satu baris	^X	Keluar dari <i>nano</i>

2.4.2. *Filesystem Hierarchy Standard (FHS)*

pada saat distro *linux* diinstall ke komputer, kita akan menemukan direktori – direktori yang secara *default* dibuat oleh *linux*. *Directory* tersebut dibuat berdasarkan *FHS (Filesystem Hierarchy Standard)*. Tujuannya agar dapat mendukung interoperabilitas aplikasi, program administrasi *system*, program pengembangan, skrip dan dapat menyatukan dokumentasi dari *system* ini. Dengan adanya *FHS (Filesystem Hierarchy Standard)* ini, pengguna dan pengembang memiliki pedoman direktori *standard* apa yang dibutuhkan untuk meracik sebuah distribusi *linux* yang operasional. Juga *file* dan pustaka, masing – masing letaknya dimana, dipandu oleh *standard* ini.

1. */ (root folder)*

Menduduki posisi paling puncak di dalam hirarki, direktori ini dilambangkan dengan tanda *slash (/)*. Direktori ini membawahi semua direktori penting lainnya. Sehingga penulisan direktori lainnya selalu menggunakan tanda *slash* di depannya, yang menunjukkan bahwa direktori tersebut berada di bawah *root*.

2. */bin*

Direktori ini berisi perintah dasar yang dibutuhkan oleh *system* maupun *user*. Sebagian perintah dasar yang bisa dijalankan disimpan dalam direktori ini.

3. */boot*

Direktori ini berisi program dan data yang dibutuhkan pada saat melakukan proses *booting* (menjalankan) *system*.

4. */dev*

Direktori ini berisi direktori tempat *file device*.

5. */etc*

Direktori ini berisi *file* konfigurasi dari *system*.

6. */home*

Direktori ini adalah tempat untuk menyimpan data *user*. Setiap *user* yang terdaftar akan secara otomatis dibuatkan direktori */home*.

7. */lib*

Direktori ini berisi *file – file library* dari aplikasi yang ada di *system*, kadangkala suatu *library* digunakan oleh beberapa aplikasi secara bersama.

8. */media*

Saat anda memasang *flashdisk* ke komputer, anda bisa menemukan direktori *flashdisk* di */media*, karena direktori ini akan berisi media yang akan dibongkar pasang di komputer anda, seperti CD (compact disk) ROM (*random access memory*), *floppy disk*, *harddisk* eksternal dan sebagainya.

9. */mnt*

Direktori tempat pengaitan *system* sementara.

10. */opt*

Direktori yang berisi aplikasi tambahan yang kita *install* ke dalam *system*.

11. */proc*

Direktori yang berisi *filesystem* untuk menjalankan proses.

12. */root*

Home directory untuk *user root*.

13. */sbin*

Direktori yang berisi program *binary* yang dibutuhkan untuk menjalankan dan memperbaiki *system*, biasanya aplikasi yang ada hanya bisa dijalankan oleh *administrator* atau *root*.

14. */temp*

Direktori yang berisi tempat penyimpanan temporer.

15. */usr*

Direktori yang berisi program – program yang dapat diakses oleh *user*, program *source code*. Di dalam direktori ini ada *sub directory* */usr/bin* dan */usr/sbin* yang menyimpan aplikasi *executable* yang fungsinya sama dengan *file – file* di direktori */bin* dan */sbin*.

16. */var*

Direktori yang berisi untuk menyimpan informasi proses, seperti *system history*, *access logs*, dan *error logs*.



Gambar 2.4 *Ubuntu Server*

2.5. *ISPconfig*

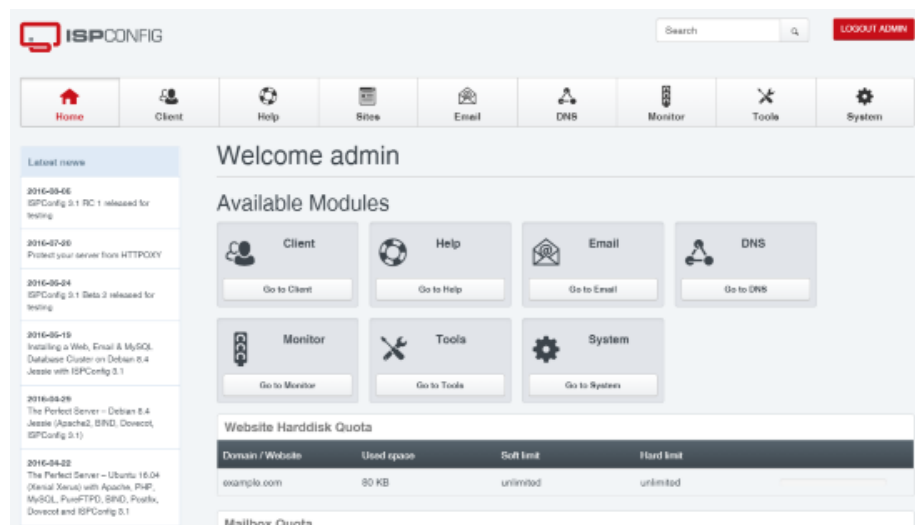
ISP (*Internet Service Provider*) *config* adalah sebuah OS *hosting control panel* untuk *linux*, Berlisensi dan di kembangkan oleh perusahaan *ISPConfig* UG pada 2005 saat musim gugur dan di mulai oleh perusahaan jerman yaitu *Projekfarm GmbH* .

Fungsinya sendiri memungkinkan *user* atau *administrator* untuk mengelola sebuah *website*, untuk DNS (*Domain Name System*) dan alamat *email* di akses secara antarmuka melalui *web*. Adapun tingkatan loginnya adalah *Administrator*, *Reseller*, *Email User*, dan *Client*.

Berikut untuk fitur *ISP Config* :

1. Manajemen *Server* untuk *Apache* dan *nginx*
2. Mangelola *server* satu atau lebih melalui *control panel*
3. *Virtual server* seperti *OpenVZ*

4. *Checking statistic dengan Webalizer*
5. *Memungkinkan seorang user menggunakan email*
6. *Penghubung seorang user ke www*
7. *ISP juga dapat memproteksi virus bagi pelanggan.*
8. *Memangement DNS Server seperti Bind, Power DNS dan lainnya.*



Gambar 2.5 ISPconfig

2.6. Apache Web Server

Apache HTTP (Hyper Text Transfer Protocol) Server (biasanya hanya disebut *Apache*) umumnya dikenal sebagai *server web* HTTP terpopuler di dunia. Ini cepat dan aman dan menjalankan lebih dari setengah dari semua *server web* di seluruh dunia. Apache juga perangkat lunak bebas, didistribusikan oleh *Apache Software Foundation* yang mempromosikan berbagai teknologi *open source advanced web technologies*. *Server web Apache* menyediakan berbagai fitur lengkap, termasuk CGI (*Computer*

Generated Imagery), *SSL (Secure Socket Layer)*, dan *domain virtual* Ini juga mendukung *plug-in* modul untuk diperpanjang.

Meskipun *Apache* pada awalnya dirancang untuk lingkungan *Unix*, hampir semua instalasi (lebih dari 90%) berjalan di *Linux*. Namun, ini juga tersedia untuk sistem operasi lain seperti *Windows*. Catatan: *Apache* memiliki *server* lain yang disebut *Apache Tomcat* yang berguna untuk *Java Servlets*.

Server, pada umumnya, adalah komputer jarak jauh yang melayani *file* untuk meminta klien. *Server web*, kemudian, adalah lingkungan tempat sebuah situs *web* berjalan, atau lebih baik lagi, komputer melayani situs *web*. Ini benar tidak peduli apa yang diberikan oleh *server web* atau bagaimana pengirimannya (*file* HTML untuk halaman *web*, *file* FTP, dll.), Atau perangkat lunak yang digunakan (misalnya *Apache*, *HFS*, *FileZilla*, *nginx*, *lighttpd*).

Server web HTTP adalah *server web* yang mengirimkan konten melalui HTTP, atau *Hypertext Transfer Protocol*, versus yang lain seperti FTP. Misalnya, ketika Anda pergi ke *Lifewire.com* di *browser web* Anda, akhirnya Anda menghubungi *server web* yang menghosting situs *web* ini sehingga Anda dapat berkomunikasi dengannya untuk meminta halaman *web* (yang telah Anda lakukan untuk melihat halaman ini).

Mengapa menggunakan *Apache HTTP Server*?

Ada sejumlah manfaat dari *Apache HTTP Server*. Yang paling menonjol adalah sepenuhnya bebas untuk penggunaan pribadi dan komersial, jadi Anda tidak perlu khawatir perlu membayarnya. Biaya satu kali pun kecil

pun tidak ada. *Apache* juga merupakan *software* yang andal dan sering diupdate karena masih aktif dipelihara. Hal ini penting saat mempertimbangkan *server web* yang akan digunakan. Anda ingin satu yang tidak hanya akan terus memberikan fitur baru dan lebih baik, tetapi juga sesuatu yang akan terus diperbarui untuk memberikan perbaikan keamanan dan perbaikan kerentanan.

Sementara *Apache* adalah produk gratis dan terupdate, namun tidak berhemat pada fitur. Sebenarnya, ini adalah salah satu *server web* HTTP yang paling banyak diisi fitur yang tersedia, yang merupakan alasan lain mengapa begitu populer.

Modul digunakan untuk menambahkan lebih banyak fungsi ke perangkat lunak. Otentikasi *password* dan sertifikat *digital* didukung. Anda dapat menyesuaikan pesan kesalahan. satu instalasi *Apache* bisa mengantarkan beberapa situs dengan kemampuan *virtual hosting*-nya; modul *proxy* tersedia; Ini mendukung SSL dan TLS (*transport layer security*), dan kompresi GZIP untuk mempercepat halaman *web*.

Berikut adalah beberapa fitur lain yang terlihat di *Apache*:

1. IPv6 (*internet protocol version 6*)
2. XML (*extensible markup language*)
3. FTP
4. *Perl*, *Lua*, dan *PHP* (*hypertext preprocessor*)
5. *Bandwidth throttling*
6. WebDAV (*web distributed authoring and versioning*)

7. Penyeimbang beban
8. HTTP / 2
9. *.htaccess*
10. *Mode Multiple Request Processing* (MPM)
11. Penulisan ulang URL (*uniform resource locator*)
12. Pelacakan sesi
13. Geolokasi berdasarkan alamat IP (*internet protocol*)

2.7. DNS (*Domain Name System*)

DNS adalah kependekan dari *Domain Name System* (DNS *server*), yaitu nama sebuah sistem *database* yang berguna untuk memenuhi kebutuhan komputer, layanan/sumber daya yang terhubung ke dalam jaringan *internet*/jaringan komputer pribadi. Atau definisi lainnya adalah merupakan sistem *database* yang terdistribusi, digunakan sebagai pencarian nama komputer di dalam jaringan yang menggunakan TCP/IP (*Transmission Control Protocol/Internet Protocol*). DNS memiliki kelebihan ukuran *database* yang tak terbatas serta mempunyai performa cukup baik. Fungsi – Fungsi DNS antara lain:

1. Menerjemahkan nama *host* (*hostnames*), jadi nomor IP *address* atau sebaliknya, sehingga nama – nama tersebut mudah diingat oleh para pengguna *internet*.
2. Memberikan suatu informasi mengenai suatu *host* kepada seluruh jaringan *internet*. DNS mempunyai keunggulan misalnya seperti: DNS

sangat mudah sebab *user* tak lagi direpotkan untuk mengingat IP (IP *address*) sebuah komputer/pc cukup *host name*. Konsisten, IP (IP *address*) sebuah komputer boleh saja berubah akan tetapi *host name* tidak boleh berubah.

2.8. DHCP (*Dinamyc Host Configuration Protocol*)

DHCP *Server* adalah kependekan dari *Dinamyc Host Configuration Protocol*, yaitu suatu layanan yang secara otomatis memberikan nomor IP kepada komputer yang memintanya. PC/komputer yang memberikan no IP inilah yang disebut dengan DHCP *server*, sedangkan komputer yang meminta atau merequest disebut dengan DHCP *Client*. Fungsi – Fungsi DHCP antara lain :

1. Mempunyai fungsi utama yaitu mendistribusikan IP (IP *address*) secara otomatis ke setiap *client*/pengguna yang terhubung dengan jaringan komputer.
2. Memberikan kemudahan untuk *network administrator*, ketika dalam mengelola jaringan komputer, sebab alokasi IP dapat ditentukan secara otomatis.
3. DHCP *server* selain bisa memberikan nomer IP *address* secara dinamik, bisa juga bisa memberikan IP *address* secara statis kepada *client*/pengguna yang terhubung kepada jaringan komputer.
4. Dapat memberikan suatu kemudahan dalam proses komunikasi data antar komputer/PC.

2.9. FTP (*File Transfer Protocol*)

FTP adalah kependekan dari “*File Transfer Protocol*” yaitu suatu protokol yang memiliki fungsi untuk tukar – menukar *file* di dalam suatu *network*/jaringan komputer yang menggunakan TCP koneksi bukan UDP (*User Datagram Protocol*). FTP *server* adalah *server* yang menjalankan *software*/perangkat lunak yang memiliki fungsi sebagai yang memberikan layanan tukar – menukar *file* yang dimana *server* tersebut selalu siap memberikan layanan *File Transfer Protocol* apabila mendapat *request* dari FTP *client*. Beberapa fungsi FTP antara lain :

1. Bertujuan untuk *sharing* data.
2. Untuk menyediakan *indirect* ataupun *implicit remote* komputer.
3. Bertujuan menyediakan tempat penyimpanan bagi pengguna/*user*.
4. Untuk menyediakan *transfer* data,*file* yang *reliable* serta efisien.

FTP *client* adalah Komputer yang meminta/merequest koneksi ke FTP *server* bertujuan untuk tukar – menukar *file*. Jika sudah terhubung dengan FTP *server*, maka *client* dapat melakukan unggah, menamai, mengunduh, menghapus, dan sebagainya sesuai dengan *permission* yang telah diberikan oleh FTP *server*.

2.10. UML (*Unified Modeling Language*)




UML(*Unified Modeling language*) adalah sebuah bahasa yang digunakan untuk menentukan visualisasi, kontruksi dan mendokumentasikan artifact dapat berupa model, deskripsi atau perangkat lunak. UML ini







berfungsi untuk membantu para *developer* untuk menggambarkan alur dari sebuah sistem yang akan dibangun, gambaran mengenai alur system tersebut akan terwakili oleh symbol – simbol yang ada dalam diagram – diagram.

2.11. Use Case Diagram

Use case pada dasarnya merupakan unit fungsionalitas koheren yang diekspresikan sebagai transaksi – transaksi yang terjadi antara *actor* dan sistem. Kegunaan *use case* sesungguhnya adalah untuk mendefinisikan suatu bagian perilaku sistem yang bersifat koheren tanpa perlu menyingkap struktur internal sistem / perangkat lunak yang sedang dikembangkan.

Tabel 2.2 *Use Case Diagram*



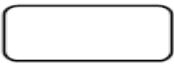
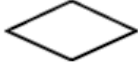


	Aktor : Menspesifikasi himpunan peran yang pengguna domain akan ketika berinteraksi dengan <i>use case</i> .
	<i>Dypendency</i> : Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri (<i>independent</i>) akan mempengaruhi elemen yang bergantung padanya elemen yang tidak mandiri (<i>indenpendent</i>)
	<i>Generalization</i> : Hubungan dimana objek anak (<i>descendent</i>) berbagi perilaku dan struktur data dari objek yang ada di atasnya objek (<i>ancestor</i>)

	<i>Include</i> : Menspesifikasikan bahwa <i>use case</i> sumber secara eksplisit.
	<i>Extend</i> : Menspesifikasikan bahwa <i>use case</i> sumber pada suatu titik yang diberikan.
	<i>System</i> :Menspesifikasikanpaket yang menampilkan sistem secara terbatas.
	<i>Use Case</i> : Deskripsi dari urutan aksi – aksi di tampilan sistem yang menghasilkan suatu hasil yang terukur bagi suatu aktor.
	<i>Collaboration</i> :Interaksi aturan – aturan dan elemen yang bekerjasama untuk menyediakan perilaku yang lebih besar dari jumlah elemen – elemennya (sinergi)
	<i>Note</i> : Elemen fisik yang eksis saat aplikasi di jalankan dan mencerminkan suatu sumber daya komputerisasi.

2.12. Activity Diagram

Activity diagram sesungguhnya merupakan bentuk khusus dari *state machine* yang bertujuan memodelkan komputasi – komputasi dan aliran – aliran kerja yang terjadi dalam sistem / perangkat lunak yang sedang dikembangkan.

Tabel 2.3 *Activity Diagram*

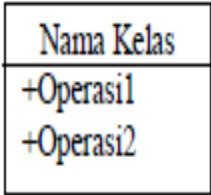
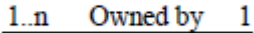

Simbol	Keterangan
	Initial :Titik awal, untuk memulai suatu aktivitas.
	Final :Titik akhir, untuk mengakhiri aktivitas.
	Activity :Menandakan sebuah aktivitas.
	Decision :Pilihan untuk mengambil keputusan.
	Fork/Join : Digunakan untuk menunjukkan kegiatan yang dilakukan secara paralel atau untuk menggabungkan dua kegiatan paralel menjadi satu.
	Rake :Menunjukkan adanya dekomposisi



2.13. Class Diagram

Diagram kelas atau *class diagram* menunjukkan interaksi antarkelas dalam sistem. Diagram kelas mengandung informasi dan tingkah laku segala sesuatu yang berkaitan dengan informasi tersebut. Adapun kegunaan dari class diagram adalah sebagai berikut :

1. Mengelompokkan objek – objek menjadi kelas – kelas berarti mengabstraksikan masalah yang sedang dihadapi.
2. Definisi – definisi *common* (seperti nama kelas dan atribut) cukup disimpan sekali per instan kelas (objek).

Tabel 2.4 *Class Diagram*


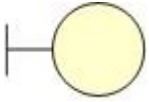
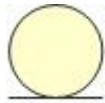
Simbol	Keterangan
	<p>Class : blok – blok pembangun pada pemrograman berorientasi objek. Sebuah class digambarkan sebagai sebuah kotak yang terbagi atas 3 bagian. Bagian atas adalah bagian nama dari <i>class</i>. Bagian tengah mendefinisikan <i>property</i>/atribut <i>class</i></p>
	<p><i>Assosiation</i> : sebuah <i>relationship</i> paling umum antara 2 <i>class</i>, dan dilambangkan oleh sebuah garis yang menghubungkan antara 2 <i>class</i>. Garis ini bisa melambangkan tipe – tipe</p>
	<p><i>Composition</i> : bagian dari class yang lain, maka <i>class</i> tersebut memiliki relasi <i>Composition</i> terhadap <i>class</i> tempat dia bergantung tersebut. Sebuah <i>relationship composition</i> digambarkan sebagai garis dengan ujung berbentuk jajaran genjang berisi/solid.</p>






	<i>Dependency</i> : digunakan untuk menunjukkan operasi pada suatu <i>class</i> yang menggunakan <i>class</i> yang lain. Sebuah <i>dependency</i> dilambangkan sebagai sebuah panah bertitik – titik.
	<i>Aggregation</i> : keseluruhan bagian <i>relationship</i> dan biasanya disebut sebagai relasi.

2.14. Sequence Diagram

Sequence diagram / diagram sekuen menggambarkan kelakuan / perilaku objek pada *use case* dengan mendeskripsikan waktu hidup objek dan *message* yang dikirimkan dan diterima antar objek. Oleh karena itu untuk menggambar diagram sekuen maka harus diketahui objek – objek yang terlibat dalam sebuah *use case* beserta metode – metode yang dimiliki kelas yang diinstansiasi menjadi objek itu.

Tabel 2.5 *Sequence Diagram*

Simbol	Keterangan
	Aktor :Menggambarkan orang yang berinteraksi dengan sistem.
	<i>Boundary Class</i> Menggambarkan sebuah penggambaran dari <i>form</i> .
	<i>Entity Class</i> Menggambarkan hubungan kegiatan yang akan dilakukan.

	<i>Control Class</i> Menggambarkan penghubung antara <i>boundary</i> dengan tabel.
	<i>Message</i> digambarkan dengan garis berpanah, yang menunjukkan arah <i>message</i>
	Garis Hidup :Menyatakan kehidupan suatu objek
	Waktu aktif :Menyatakan objek dalam keadaan aktif dan berinteraksi.
	Stimulus: Menyatakan suatu objek mengirimkan pesan untuk menjalankan operasi yang ada pada objek lain.