

Maaf bu baru isinya saja, belum dirapihkan. Nanti mau dimasukan ke latex biar langsung ada caption untuk setiap gambar dan table. Kami sudah minta format nya ke liyana dan pas kami lihat itu sama dengan yang latex jadi kami mau memasukan isi data ini ke file latex setelah beres observasi ke SAMICI. Terimakasih bu sebelumnya.

BAB I APLIKASI BANK SAMPAH

Pertumbuhan ekonomi masyarakat Indonesia dari tahun ke tahun semakin meningkat diikuti dengan pertumbuhan penduduk. Hal tersebut semakin terasa dampaknya terhadap lingkungan yaitu manusia cenderung merusak lingkungan demi mempertahankan hidupnya. Kualitas lingkungan secara terus menerus semakin menurun sehingga menimbulkan permasalahan degradasi lingkungan pada kehidupan masyarakat. Salah satu permasalahan lingkungan yang masih menjadi problematika di perkotaan yaitu pengelolaan sampah.

Pengelolaan sampah adalah kegiatan yang sistematis, menyeluruh, dan berkesinambungan yang meliputi pengurangan dan penanganan sampah. Sehingga pengelolaan pada kawasan perkotaan, dewasa ini dihadapkan kepada berbagai permasalahan yang cukup kompleks. Permasalahan-permasalahan tersebut meliputi tingginya laju timbunan sampah, kepedulian masyarakat (human behaviour) yang masih sangat rendah serta masalah pada kegiatan pembuangan akhir sampah (final disposal)

Kurangnya kepedulian masyarakat dalam menjaga kebersihan sampah dapat merugikan banyak pihak baik itu merugikan dirinya sendiri maupun masyarakat sekitar. Sampah yang sering kali menumpuk di suatu tempat dan lama kelamaan akan semakin banyak sehingga menyebabkan banyak masalah, dan hal itu perlu adanya kepedulian dari masyarakat sekitarnya dan sebuah sistem yang dapat mengelola sampah menjadi sesuatu yang berguna.

Selain hal tersebut di dalam masyarakat perkotaan terdapat budaya konsumtif yang mempengaruhi dalam peningkatan kualitas

dan jenis sampah. Sehingga dalam pengelolaan sampah tidak akan dapat dipisahkan dengan campur tangan negara dan berbagai sektor yang ada di dalam masyarakat termasuk dunia usaha. Selain itu peran dari masyarakat yang merupakan jejaring atau komunitas pembuang sampah juga mempunyai andil besar dalam pengelolaan sampah dalam hal ini adalah proses daur ulang untuk dapat dimanfaatkan kembali. Sehingga dalam pengelolaan sampah merupakan bagian dari pelayanan publik yang harus diatur dalam regulasi yang diharapkan akan memberikan kenyamanan di dalam kehidupan masyarakat warga sehari-hari.

Di Indonesia sebenarnya terdapat beberapa peraturan perundang-undangan yang mempunyai korelasi maupun berkaitan langsung dengan pengelolaan sampah yaitu Undang-Undang No. 32 Tahun 2009 tentang Perlindungan dan Pengelolaan Lingkungan Hidup, Undang-undang Nomor 32 tahun 2004 tentang Pemerintahan Daerah diganti dengan UU No. 23 Tahun 2014 tentang Pemerintahan Daerah, UU No. 18 Tahun 2008 tentang Pengelolaan Sampah dan beberapa peraturan daerah yang sudah dibentuk oleh pemerintah daerah baik di tingkat Kabupaten atau Kota seperti di Peraturan Daerah Kota Surakarta No. 3 tahun 2010 Tentang Pengelolaan Sampah. Sanksi-sanksi yang terdapat dalam peraturan terutama yang menyangkut pengelolaan sampah tidak memberikan efek jera bagi masyarakat yang tidak melakukan pengelolaan sampah dengan berwawasan lingkungan sehingga perlu dikaji mengenai efektifitas sanksi dalam penegakan hukum dalam pengelolaan sampah. Selain itu peran pemerintah daerah juga sangat penting dalam mengeluarkan kebijakan terhadap pengelolaan sampah. Apabila daerah mampu

mengelola sampahnya dengan baik maka pelaksanaan terhadap prinsip Good Environmental Governance sudah dapat dikatakan terpenuhi. Oleh karena itu, akan dirancang sebuah Aplikasi pengelolaan sampah yang akan diintegrasikan dengan Aplikasi Bank Sampah yang akan dirancang pada proyek ini.

Dalam perancangan aplikasi ini menggunakan sebuah kerangka kerja yaitu *codeigniter* yang sudah banyak dipakai dalam pengembangan *website*, yang ringan dan cepat dalam membangun pemrograman website dinamis. Berbeda dengan cms yang memang dapat di unduh secara gratis namun kurang dapat dikembangkan karena masih bersifat statis sehingga perancangan *website* Aplikasi Bank Sampah ini menggunakan kerangka kerja yang dapat dikembangkan dengan dinamis.

BAB II LANDASAN TEORI

2.1 Sistem

Sistem adalah suatu kelompok dari komponen-komponen yang saling berhubungan dengan satu sama lain dengan tujuan yang sama untuk mencapai tujuan tertentu. Sedangkan informasi adalah berupa data yang diolah menjadi bentuk lebih berguna dan berarti bagi pemakainya.[4] Sehingga dapat disimpulkan bahwa system informasi adalah suatu komponen yang saling berhubungan yang bertujuan mengumpulkan, mengubah, dan menyebarkan informasi dalam sebuah organisasi.



Gambar Sistem

Sumber : <http://www.naesys.com/>

Sistem terdiri dari bagian-bagian yang saling berkaitan yang beroperasi bersama untuk mencapai beberapa sasaran atau maksud (Budi Sutedjo, 2006:11). Sistem terdiri dari elemen-elemen yang berinteraksi untuk mencapai tujuan tertentu suatu sistem mempunyai karakteristik atau sifat-sifat tertentu. Terdapat dua kelompok pendekatan dalam mendefinisikan sistem, yaitu yang menekankan pada prosedurnya dan yang menekankan pada komponen atau elemennya.

Suatu sistem adalah suatu jaringan kerja untuk melakukan suatu kegiatan atau untuk menyelesaikan suatu sasaran tertentu. Sedangkan pengertian prosedur itu sendiri menurut Richard F. Neuschel, prosedur suatu urutan operasi klerikal (tulis menulis), biasanya melibatkan beberapa orang dalam satu atau lebih departemen, yang diterapkan untuk menjamin penanganan yang seragam dari transaksi-transaksi bisnis yang terjadi.

2.1.1 Karakteristik Sistem

Suatu sistem memiliki karakteristik atau sifat-sifat tertentu, yaitu (Sutabri, 2004:12): 153

1. Komponen-komponen (*Components*)

Suatu sistem terdiri dari sejumlah komponen yang sering disebut dengan subsistem yang saling berinteraksi, yang artinya saling bekerja sama membentuk satu kesatuan. Komponen-komponen sistem atau elemen-elemen sistem dapat berupa suatu subsistem atau bagian-bagian dari sistem. Setiap subsistem mempunyai

sifat-sifat dari sistem untuk menjalankan suatu fungsi tertentu dan mempengaruhi proses sistem secara keseluruhan.

2. Batas sistem (*Boundary*)

Batas sistem merupakan daerah yang membatasi antara suatu sistem dengan sistem yang lainnya atau dengan lingkungan luarnya. Batas sistem ini memungkinkan suatu sistem dipandang sebagai satu kesatuan. Batas suatu sistem menunjukkan ruang lingkup (scope) sistem itu sendiri.

3. Lingkungan luar sistem (*Environments*)

Lingkungan luar dari suatu sistem adalah apapun diluar batas dari sistem yang mempengaruhi operasi sistem. Lingkungan luar sistem dapat bersifat menguntungkan dan dapat juga bersifat merugikan sistem tersebut.

4. Penghubung sistem (*Interface*)

Penghubung merupakan media penghubung antara subsistem dengan subsistem lainnya. Melalui penghubung ini memungkinkan sumber-sumber daya mengalir dari satu subsistem ke subsistem yang lainnya.

5. Masukan sistem (*Input*)

Masukan yaitu energi yang dimasukkan ke dalam sistem, dimana dapat berupa masukan perawatan (*maintenance input*) dan masukan sinyal (*signal input*). Masukan perawatan adalah energi yang di inputkan supaya sistem tersebut dapat beroperasi, sedang masukan sinyal adalah energi yang diproses untuk didapatkan keluaran.

6. Keluaran sistem (*Output*)

Keluaran yaitu hasil dari energi yang diolah dan diklasifikasikan menjadi keluaran yang berguna dan sisa pembuangan.

7. Pengolah sistem

Suatu sistem dapat mempunyai suatu bagian pengolah yang akan merubah *input* menjadi *output*.

8. Sasaran sistem (*Objective*)

Suatu sistem pasti mempunyai tujuan (*goal*) atau sasaran (*objective*). Apabila suatu sistem tidak mempunyai sasaran, maka operasi sistem tidak ada gunanya.

2.1.2 Klasifikasi Sistem

Sistem dapat diklasifikasikan dari berbagai sudut pandang, diantaranya adalah sebagai berikut (Sutabri, 2004:14):

1. Sistem abstrak dan sistem fisik.

Sistem abstrak adalah suatu sistem yang berupa ide-ide pemikiran atau ide-ide yang tidak tampak secara fisik, misalnya teologi yaitu sistem yang berupa pemikiran-pemikiran hubungan antara manusia dengan Tuhan.

Sedangkan sistem fisik adalah suatu sistem yang berupa pemikiran atau ide-ide yang nyata atau yang ada secara fisik. Misalnya sistem komputer, sistem akuntansi, sistem produksi dan lain sebagainya.

2. Sistem alamiah dan sistem buatan manusia

Sistem alamiah adalah sistem yang terjadi melalui proses alam, tidak dibuat manusia. Misalnya: sistem perputaran

bumi. Sistem buatan manusia yang melibatkan interaksi antara manusia dengan mesin disebut dengan human-machine system atau ada yang menyebut dengan man-machine system.

3. Sistem tertentu dan sistem tak tentu

Sistem tertentu beroperasi dengan tingkah laku yang sudah dapat diprediksi. Contohnya: sistem komputer. Sistem tak tentu adalah sistem yang kondisi masa depannya tidak dapat diprediksi karena mengandung unsur probabilitas.

4. Sistem tertutup dan sistem terbuka

Sistem tertutup merupakan sistem yang tidak berhubungan dan tidak berpengaruh dengan lingkungan luarnya. Sedangkan sistem terbuka adalah sistem yang berhubungan dan terpengaruh dengan lingkungan.

Menurut Scott (1996) sistem terdiri dari unsur-unsur seperti masukan (*input*), pengolahan (*processing*), keluaran (*output*) (Hanif Al Fatta 2007:4). Menurut McLeod (2004) sistem adalah sekelompok elemen-elemen yang terintegrasi dengan tujuan yang sama untuk mencapai tujuan (Yakub, 2012:1).

Sedangkan menurut Jogiyanto (1999) terdapat dua kelompok pendekatan sistem didalam mendefinisikan sistem yaitu pendekatan pada prosedur dimana sistem adalah suatu jaringan kerja dari prosedur-prosedur yang saling berhubungan, terkumpul bersama-sama untuk melakukan suatu kegiatan atau untuk tujuan tertentu, dan pendekatan pada komponen-komponen atau elemen-elemen, pendekatan pada komponen dianggap lebih mudah dalam mempelajari sistem untuk tujuan dan perancangan system (Yakub, 2012:1).

2.2 Sistem Informasi

Menurut Whitten, Bentley, Dittman “ Sistem informasi adalah pengaturan manusia, data, prosesproses, antarmuka yang saling berinteraksi untuk mendukung kebutuhan penyelesaian masalah dan pengambilan keputusan oleh manajemen dan para pengguna.” (1) “Sistem informasi adalah suatu sistem yang menggunakan teknologi informasi untuk mengambil, menyalurkan, menyimpan, mengambil kembali, memanipulasi, atau menampilkan informasi yang digunakan untuk proses bisnis” (2) Sedangkan menurut Wilkinson (Wilkinson, 1993:14), “Sistem Informasi adalah suatu kerangka kerja dengan mana sumberdaya (manusia,komputer) dikoordinasikan untuk mengubah masukan (data) menjadi keluaran (informasi), guna mencapai sasaran perusahaan”.

Dapat disimpulkan bahwa sistem informasi adalah kumpulan elemen yang saling berhubungan satu sama lain dengan memberikan masukan (data) untuk mendapatkan suatu keluaran (informasi) dengan fungsi untuk memproses, menyimpan, mendistribusikan informasi serta mendukung pengambilan keputusan.

Informasi adalah data yang telah diolah menjadi sebuah bentuk yang berarti bagi penerimanya dan bermanfaat dalam pengambilan keputusan saat ini atau mendatang. Sedangkan menurut McLeod dalam informasi sebagai data yang telah diolah menjadi bentuk yang lebih berarti bagi penerimanya (Ladjamudin, 2005:5-6).

2.2.1 Kualitas Informasi

Informasi yang baik adalah informasi yang berkualitas (Sutabri, 2004:25), informasi yang berkualitas ditentukan oleh beberapa hal, yaitu :

1. Akurat (*accurate*)

Informasi harus bebas dari kesalahan-kesalahan dan tidak menyesatkan, informasi harus jelas mencerminkan maksudnya.

2. Tepat waktu (*time lines*)

Informasi yang dihasilkan atau dibutuhkan tidak boleh terlambat, karena nantinya tidak mempunyai nilai yang baik, sehingga apabila dijadikan dasar dalam pengambilan keputusan akan berakibat fatal atau kesalahan pengambilan keputusan dan tindakan.

3. Relevan (*relevance*)

Berarti informasi tersebut mempunyai manfaat bagi pemakainya. Informasi merupakan *Output* dari data yang telah diolah sedemikian rupa hingga menjadi suatu bentuk yang berguna dan telah mempunyai arti. *Output* yang tidak memiliki tiga unsur diatas (Akurasi, Tepat Waktu, Relevan) dapat dikatakan sebagai informasi yang berguna, tetapi merupakan sampah.

Sistem informasi bukan merupakan hal yang baru. Yang baru adalah komputerisasinya. Sebelum ada komputer, Teknik penyaluran informasi yang memungkinkan manajer merencanakan serta

mengendalikan operasi yang telah ada. Komputer menambahkan satu atau dua dimensi, seperti kecepatan, ketelitian dan penyediaan data dengan volume yang lebih besar yang memberikan bahan pertimbangan yang lebih banyak untuk mengambil keputusan.

Sistem informasi sistem yang didalamnya mempertemukan kebutuhan pengolahan transaksi harian yang mendukung fungsi operasi organisasi yang bersifat manajerial dengan kegiatan strategi dari suatu organisasi untuk dapat menyediakan kepada pihak luar tertentu dengan laporan-laporan yang diperlukan (Sutabri, 2004:35).

Sistem Informasi Sistem adalah kumpulan elemen yang saling berhubungan satu sama lain yang membentuk satu kesatuan dalam usaha mencapai suatu tujuan [5]. Sedangkan dalam buku yang berbeda [3], sistem adalah sekumpulan elemen atau subsistem yang saling bekerjasama atau yang dihubungkan dengan cara tertentu sehingga membentuk satu kesatuan untuk melaksanakan suatu fungsi guna mencapai suatu tujuan.

Informasi adalah hasil pemrosesan data yang diperoleh dari setiap elemen sistem tersebut menjadi bentuk yang mudah dipahami dan merupakan pengetahuan yang relevan yang dibutuhkan oleh orang untuk menambah pemahamannya terhadap fakta-fakta yang ada [5]. Sedangkan pendapat yang berbeda [3], informasi merupakan hasil pengolahan data sehingga menjadi bentuk yang penting bagi penerimanya dan mempunyai kegunaan sebagai dasar dalam pengambilan keputusan yang dapat dirasakan akibatnya secara

langsung saat itu juga atau secara tidak langsung pada saat mendatang.

Sistem informasi adalah kumpulan elemen yang saling berhubungan satu sama lain yang membentuk satu kesatuan untuk mengintegrasikan data, memproses dan menyimpan serta mendistribusikan informasi [5]. Dapat pula diartikan bahwa sistem informasi adalah suatu kumpulan dari komponen-komponen dalam perusahaan atau organisasi yang berhubungan dengan proses penciptaan dan pengaliran informasi [3].

2.3 Integrasi Sistem

Integrasi sistem didefinisikan sebagai suatu rekayasa dalam proses menyatukan subsistem komponen kedalam satu sistem (agregasi dari sub sistem yang bekerja sama sehingga sistem mampu memberikan fungsi menyeluruh) dan memastikan bahwa subsistem berfungsi bersama sebagai suatu sistem, dan dalam teknologi informasi sebagai proses menghubungkan bersama berbagai sistem yang terkomputasi dan aplikasi perangkat lunak secara fisik atau fungsional untuk bertindak sebagai keseluruhan yang terkoordinasi.



Gambar Integrasi Sistem

Sumber : coins.newbium.com

Sistem integrasi memanfaatkan berbagai Teknik dalam mengintegrasikan dua sistem seperti jaringan computer, integrase aplikasi *enterprise*, manajemen proses bisnis atau pengguna pemrograman.

Didunia modern yang terhubung oleh internet integrase sistem sangat memiliki peran penting. Semakin banyak sistem dirancang untuk terhubung, baik dalam sistem yang akan dibangun maupun ke sistem yang sudah digunakan.

Integrasi Sistem Informasi sering disebut *Enterprise Information System*, yaitu sebuah platform teknologi yang memungkinkan organisasi mengintegrasikan dan mengkoordinasikan proses bisnis. Sedangkan Sandoe (2001) mengatakan bahwa integrasi Sistem Informasi bertujuan menggabungkan Sistem Informasi yang tadinya terpisah dengan tujuan sebuah sumber daya informasi yang lebih komplit dan menyeluruh bagi sebuah organisasi.[5]

2.4 Bank

Bank berasal dari bahasa Prancis “Banque” atau dalam bahasa Italia disebut Bianco yang berarti peti, meja atau tempat menyimpan uang. Kata bank dalam bahasa Italia yang berarti meja memang diambil dari kata tersebut karena transaksi keuangan dalam lembaga tersebut biasa dilakukan diatas meja. Dalam berdasarkan undang-undang yang ada di Indonesia, bank memiliki maknanya tersendiri. Menurut Undang-undang Nomor 7 Tahun 1992 Tentang Perbankan , Bank adalah badan usaha yang menghimpun dana dari masyarakat dalam bentuk simpanan, dan menyalurkan kepada masyarakat dalam rangka meningkatkan taraf hidup rakyat banyak.

Pengertian Bank Yang dinyatakan Oleh Drs. H. Malayu S.P. Hasibuan, Yakni :

Bank adalah badan usaha kekayaan terutama didalam bentuk aset keuangan (financial assets) dan juga bermotifkan profit serta sosial, jadi bukan hanya mencari keuntungan saja. Bank ialah pencipta dan juga pengedar uang kartal. Pencipta serta pengedar uang kartal (uang

kertas dan juga logam) merupakan otoritas tunggal dari bank sentral (Bank Indonesia), sedangkan uang giral dapat diciptakan dengan bank umum.

Jenis jenis bank, maka dilihat dari fungsinya jenis jenis bank ada 4 yaitu :

Bank Sentral, yakni jenis bank yang bertugas untuk menerbitkan uang kertas dan juga uang logam untuk dapat dijadikan sebagai alat pembayaran yang sah di dalam suatu negara dan juga mempertahankan konversi uang yang dimaksud terhadap emas maupun perak maupun keduanya.

Bank Umum, yakni jenis bank yang bukan saja dapat untuk meminjamkan ataupun menginvestasikan berbagai jenis tabungan yang diperolehnya, namun tetapi juga dapat memberikan pinjaman dari menciptakan sendiri suatu uang giral.

Bank Perkreditan Rakyat (BPR), yaitu jenis bank yang melaksanakan kegiatan usaha dengan secara konvensional maupun yang didasarkan pada suatu prinsip syariah yang dalam kegiatannya tidak dapat memberikan jasa di dalam lalu lintas pembayaran.

Bank Syariah, yakni jenis bank yang beroperasi dengan berdasarkan prinsip bagi hasil maupun sesuai dengan kaidah ajaran islam mengenai hukum riba.

2.5 Sampah

Sampah adalah tumpukan bahan bekas dan sisa tanaman (daun, sisa sayuran, sisa buangan lain), atau sisa kotoran hewan atau benda-benda lain yang dibuang. Dalam pengertian yang luas, sampah diartikan sebagai benda yang dibuang, baik yang berasal dari alam ataupun dari hasil proses teknologi (Reksosoebroto, 1990). Menurut Wasito (1970) sampah ialah segala zat padat atau semi padat yang terbuang atau yang sudah tidak berguna, baik yang dapat membusuk atau yang tidak dapat membusuk kecuali zat-zat buangan atau kotoran yang keluar dari tubuh manusia (kotoran atau najis manusia).

Menurut Reksosoebroto (1990), bahwa penanganan sampah yang baik akan memberikan manfaat yang besar bagi kehidupan manusia dan lingkungan. Manfaat lain penanganan sampah yang baik adalah menurunkan 90% angka kehidupan lalat menurunkan 90% angka kehidupan tikus menurunkan 30% angka kehidupan nyamuk, menurunkan 70% angka kerusakan jembatan dan menurunkan 90% angka kerusakan pipa bangunan. Keuntungan pembuangan sampah yang dapat diperoleh dari pengelolaan sampah yang baik dapat dilihat dari beberapa segi yaitu: (1) Dari segi sanitasi, menjamin tempat kerja yang bersih, mencegah tempat berkembang biaknya vektor hama penyakit dan mencegah pencemaran lingkungan termasuk timbulnya pengotoran sumber air; (2) Dari segi ekonomi mengurangi biaya perawatan dan pengobatan sebagai akibat yang ditimbulkan sampah. Tempat kerja yang bersih akan meningkatkan gairah kerja dan akan

menambah produktivitas serta efisiensi pekerja, menarik banyak tamu atau pengunjung, mengurangi kerusakan sehingga mengurangi biaya perbaikan (3) Dari segi estetika, menghilangkan pemandangan tidak sedap dipandang mata menghilangkan timbulnya bau–bauan yang tidak enak, mencegah keadaan lingkungan yang kotor dan tercemar. Penanganan sampah yang baik akan memberikan manfaat yang besar bagi kehidupan manusia dan lingkungan.

2.6 Bank Sampah

Menurut Peraturan Menteri Negara Lingkungan Hidup RI Nomor 13 Tahun 2012 Tentang Pedoman Pelaksanaan Reduce, Reuse, dan Recycle Melalui Bank Sampah, bank sampah sendiri di atur dalam pasal 1 ayat 2 peraturan ini. Adapun bunyi dari pasal ini yaitu: “Bank sampah adalah tempat pemilahan dan pengumpulan sampah yang dapat didaur ulang dan/atau diguna ulang yang memiliki nilai ekonomi.”

2.7 Framework

Framework adalah kumpulan perintah atau fungsi dasar yang membentuk aturan-aturan tertentu dan saling berinteraksi satu sama lain sehingga dalam pembuatan aplikasi website, kiat harus mengikuti aturan dari *framework*.

Dengan *framework* kita tidak perlu memikirkan kode perintah/ fungsi dasar dari aplikasi website. Seperti bagaimana mengambil data dari database untuk ditampilkan. Kita hanya memikirkan apa kode SQLnya dan ditampilkan kemana? Hal

penunjang lainnya seperti koneksi database, validasi form, GUI, dan keamanan telah disediakan oleh *framework* sehingga jumlah baris kode yang kita buat jauh lebih sedikit dibandingkan jika semua kode dari kita.

Jadi keuntungan yang dapat diperoleh dari penggunaan *framework* adalah:

1. Waktu pembuatan aplikasi website kita jauh lebih singkat.
2. Kode aplikasi website menjadi lebih mudah dibaca, karena sedikit dan sifatnya pokonya. Detailnya adalah kode daari *framework* dan ini mungkin difikirkan, terjamin.
3. Website kita menjadi lebih mudah diperbaiki, karena kita tidak perlu focus kesemua komponen kode website terutama kode sistem *framework*.
4. Kita tidak perlu lagi membuat kode penunjang aplikasi website seperti koneksi database, validasi form GUI dan keamanan.
5. Fikiran kita menjadi lebih terfokus ke kode alur permasalahan website. Apa yang ditampilkan dan layanan apa saja yang diberikan dari apliakasi website tersebut.
6. Jika dikerjakan teamwork maka akan lebih terarah karena sistem akan *framework*, mengharuskan adanya keteraturan peletakan kode seperti bagian pengambilan databse terpisah dengan bagian pengaturan tampilan untuk pengunjung.

CodeIgniter dilengkapi dengan berbagai pustaka siap pakai untuk berbagai kebutuhan misalnya saja koneksi database, email, *session*, dan *cokies*, keamanan dan manipulasi gambar dan banyak lagi sehingga mempermudah pekerjaan kita. Dan jika tidak tersedia kita dapat saja menambah pustaka yang ada dengan menciptakan pustaka sendiri yang di *includekan* kedalam *framework* tersebut.

2.8 *CodeIgniter*

CodeIgniter adalah kerangka kerja aplikasi web untuk membangun aplikasi web yang sifatnya *open source* yang digunakan untuk membangun aplikasi PHP yang dinamis. *CodeIgniter* memiliki Karakteristik yang sangat fleksibel dan ringan dalam memodifikasi dan mengintegrasikan suatu website. Dalam *CodeIgniter* dapat juga digunakan pola *Model View Controller* (MVC) sehingga struktur kode yang di hasilkan lebih terstruktur dan memiliki standar yang jelas.[8]

CodeIgniter adalah salah satu jawabannya bagaimana kita membuat website tidak dari nol. Dengan *codeigniter* kita tidak perlu pusing membuat website PHP karena *framework codeigniter* ini sudah menyediakan fasilitas untuk mempercepat pembuatan website. *Codeigniter* ini menyediakan fungsi-fungsi kode yang dapat digunakan kembali dalam pembuatan website dan ini mempercepat efisiensi dalam mengembangkan aplikasi web.

Ada banyak keuntungan penggunaan *codeigniter* antara lain penghematan waktu, *script website* menjadi lebih mudah dibaca dan diupdate, mempermudah pembuatan sistem besar dan terstruktur, kode *script* kita menjadi lebih mudah dikelola dan karena adanya pengelompokan kode maka akan mempermudah pembagian kerja dalam tim jika kita membuat *website* secara gotong royong.

2.9 PHP (*Hypertext Preprocessor*)

PHP adalah singkatan dari *Hypertext Preprocessor* yang merupakan bahasa pemrograman yang bersifat *open source* yang berada didalam server yang diproses di server. PHP salah satu bahasa pemrograman yang memiliki script yang terintegrasi dengan HTML dan berada pada server (*server side HTML embedded scripting*).[9]

PHP merupakan bahasa pemrograman pelengkap HTML yang memungkinkan aplikasi web dinamis untuk pengolahan data, pemrosesan data dari *user via form*, membuat buku tamu, toko online, dan lain sebagainya. Dengan mudah PHP dapat melakukan koneksi ke database karena PHP memang dilengkapi fitur yang memungkinkan koneksi ke PHP dilakukan dengan mudah tanpa harus memusingkan (Tim EMS, 2016:55).

PHP adalah bahasa *server-side scripting* yang menyatu dengan HTML untuk membuat halaman web yang dinamis. Maksud dari *server-side scripting* adalah sintaks dan perintah-perintah yang diberikan akan sepenuhnya akan dijalankan diserver tetapi disertakan pada dokumen HTML. Pembuatan web ini merupakan kombinasi

antara php sendiri sebagai bahasa pemrograman dan HTML sebagai pembangun halaman web (Bimo sunarfrihantono, ST 2002:9).



Gambar Hypertext Preprocessor

Sumber : <https://www.webhozz.com/>

Menurut Andi Pramono dan M. Syafii (2004 : 2), PHP atau kependekan dari *Hypertext Preprocessor* adalah sebuah bahasa pemrograman berbasis *web* yang mempunyai banyak keunggulan dibandingkan dengan bahasa pemrograman berbasis *web* yang lain. PHP merupakan bahasa pemrograman yang bersumber dari Perl. Sedangkan Perl merupakan pengembangan dari bahasa C.

Oleh karenanya, struktur pemrograman yang ada di PHP sama dengan yang ada di bahasa C. Melihat bahwa PHP merupakan pengembangan dari bahasa C secara tidak langsung, maka PHP mempunyai banyak sekali fitur-fitur yang dapat digunakan. Misalnya,

PHP dapat mengakses shell di Linux, mempunyai fungsi yang lengkap berhubungan dengan *networking*.

Salah satu keunggulan PHP dibanding bahasa pemrograman lainnya adalah PHP dapat diperoleh secara gratis, meskipun bukan berarti karena gratis kemampuannya menjadi pas-pasan. PHP sangat powefull. Terbukti dengan banyaknya website yang dibangun menggunakan PHP.

PHP juga terkenal lebih aman daripada bahasa pemrograman website yang lain. PHP juga sudah mendukung OOP (*Object Oriented Programming*) sehingga *maintenance* kode menjadi jauh lebih muda dibandingkan procedural.

PHP bisa berinteraksi dengan database, file dan folder, PHP ini termaksud bahasa *cross-platform*, yang mana PHP ini bisa berjalan di sistem operasi yang berbeda-beda (Windows, Linux, ataupun MAC).[10]

Berdasarkan pernyataan diatas dapat disimpulkan bahwa PHP adalah bahasa adalah sebuah bahasa pemrograman yang sangat fleksibel untuk digunakan dalam membuat sebuah website, dan juga PHP dapat dijalankan di bawah sistem operasi LINUX dan Windows.

2.10 MySQL (*My Structure Query Language*)

MySQL adalah sebuah server database *open source* yang terkenal yang digunakan berbagai aplikasi terutama untuk server atau membuat web. MySQL juga adalah sebuah implementasi dari sistem

manajemen basis data relasional (RDBMS). Keandalan suatu sistem basis data (DBMS) dapat diketahui dari cara kerja pengoptimasinya dalam melakukan proses perintah-perintah SQL yang dibuat oleh pengguna maupun program-program aplikasi yang memanfaatkannya.[10]

Berdasarkan dari pengertian diatas kesimpulan dari MySQL adalah sekumpulan sistem database yang banyak digunakan untuk pengembangan suatu aplikasi berbasis web dan bersifat open source.

MySQL merupakan software yang tergolong sebagai DBMS (*Database Management System*) yang bersifat *Open Source*. Open Source menyatakan bahwa *software* ini dilengkapi dengan *source code*. MySQL pada awalnya dibuat oleh perusahaan konsultan bernama TcX yang berlokasi di Swedia. Saat ini pengembangan MySQL berada dibawah naungan perusahaan MySQL AB (Kadir, 2008:2).

Fitur yang terdapat pada MySQL:

1. *Multipatform*

MySQL tersedia pada beberapa platform (Windows, Linux, Unix dan lain-lain).

2. Andal, cepat dan mudah digunakan MySQL tergolong sebagai database server yang andal, dapat menangani database yang besar dengan kecepatan tinggi, mendukung banyak sekali fungsi untuk mengakses database, dan sekaligus mudah untuk digunakan.

3. Jaminan keamanan akses

MySQL mendukung pengamanan database dengan berbagai kriteria pengaksesan.

4. Dukungan SQL

MySQL mendukung perintah SQL (*Structurd Query Language*). Sebagai diketahui, SQL merupakan standar dalam pengaksesan database relasional.

2.8. 1 Perkembangan MySQL

Pada bulan Mei 1996, MySQL versi 1.0 berhasil dirilis secara terbatas hanya untuk 4 orang saja. Namun di bulan Oktober pada tahun yang sama versi 3.11.0 dilepas ke publik. Namun mula-mula kode ini tidak diberikan dibawah lisensi GPL (*General Public License*), melainkan lisensi khusus. Pada tahun 1998-1999, yaitu pada versi-versi akhir 3.22, MySQL menjadi semakin populer dan dilirik orang karena kestabilan dan kecepatan yang meningkat. Jika pada versi 3.22, MySQL mulai diadopsi oleh banyak orang. Berbeda halnya dengan versi 3.23 dan 4.0 yang telah terjadi banyak peningkatan dari sisi teknologi (Sukarno, 2006:5-7).

MySQL memiliki beberapa keistimewaan, antara lain :

1. Portabilitas, MySQL dapat berjalan stabil pada berbagai sistem operasi seperti Windows, Linux, FreeBSD, Mac Os X Server, Solaris, Amiga, dan masih banyak lagi.

2. Perangkat lunak sumber terbuka. MySQL didistribusikan sebagai perangkat lunak sumber terbuka, dibawah lisensi GPL sehingga dapat digunakan secara gratis.
3. *Multi-user*. MySQL dapat digunakan oleh beberapa pengguna dalam waktu yang bersamaan tanpa mengalami masalah atau konflik.
4. *'Performance tuning'*, MySQL memiliki kecepatan yang menakjubkan dalam menangani query sederhana, dengan kata lain dapat memproses lebih banyak SQL per satuan waktu.
5. Ragam tipe data. MySQL memiliki ragam tipe data yang sangat kaya, seperti *signed / unsigned integer, float, double, char, text, date, timestamp*, dan lainlain.
6. Perintah dan Fungsi. MySQL memiliki operator dan fungsi secara penuh yang mendukung perintah Select dan Where dalam perintah (*query*).
7. Keamanan. MySQL memiliki beberapa lapisan keamanan seperti level subnetmask, nama host, dan izin akses user dengan sistem perizinan yang mendetail serta sandi terenkripsi.
8. Skalabilitas dan Pembatasan. MySQL mampu menangani basis data dalam skala besar, dengan jumlah rekaman (*records*) lebih dari 50 juta dan 60 ribu tabel serta 5 milyar baris. Selain itu batas indeks yang dapat ditampung mencapai 32 indeks pada tiap tabelnya.
9. Konektivitas. MySQL dapat melakukan koneksi dengan klien menggunakan protokol TCP/IP, Unix soket (UNIX), atau Named Pipes (NT).

10. Lokalisasi. MySQL dapat mendeteksi pesan kesalahan pada klien dengan menggunakan lebih dari dua puluh bahasa. Meski pun demikian, bahasa Indonesia belum termasuk di dalamnya.
11. Antar Muka. MySQL memiliki antar muka (*interface*) terhadap berbagai aplikasi dan bahasa pemrograman dengan menggunakan fungsi API (*Application Programming Interface*).
12. Klien dan Peralatan. MySQL dilengkapi dengan berbagai peralatan yang dapat digunakan untuk administrasi basis data, dan pada setiap peralatan yang ada disertakan petunjuk online.
13. Struktur tabel. MySQL memiliki struktur tabel yang lebih fleksibel dalam menangani ALTER TABLE, dibandingkan basis data lainnya semacam PostgreSQL ataupun Oracle

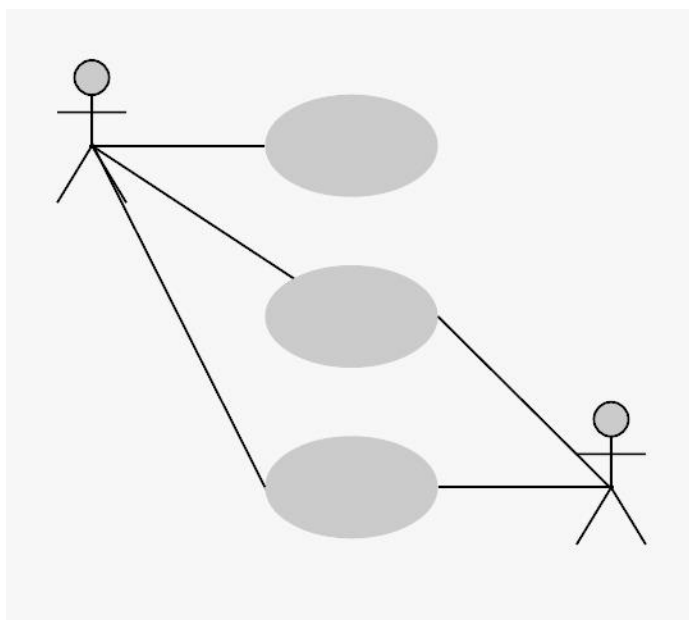
2.11 UML (*Unified Modeling Language*)

The Unified Modeling Language (UML) adalah standar de facto untuk analisis perangkat lunak berorientasi objek dan pemodelan desain.[11] UML membantu memodelkan berbagai aspek sistem melalui berbagai diagram yang di dukungnya. Setiap aspek dari suatu sistem disajikan dengan menggunakan jenis diagram UML tertentu dan satu set diagrama disebut sebagai model.[12]

UML mempunyai sejumlah elemen grafis yang bisa dikombinasikan menjadi diagram. Karena ini merupakan sebuah bahasa, UML memiliki sejumlah aturan untuk menggabungkan atau mengkombinasikan elemen-elemen tersebut.

1. Use case diagram

Use case diagram menjelaskan apa yang akan dilakukan oleh sistem yang akan dibangun dan siapa yang berinteraksi dengan sistem. *Use case diagram* menjadi dokumen kesepakatan antara *customer*, *user* dan *developer*. *User* menggunakan dokumen *use case diagram* ini untuk memahami sistem dan mengevaluasi bahwa benar yang dilakukan sistem adalah untuk memecahkan masalah yang user ajukan. *Use case diagram* memberikan gambaran statis dari sistem yang sedang dibangun dan merupakan artifak dari proses analisis (Hermawan, 2004:23).

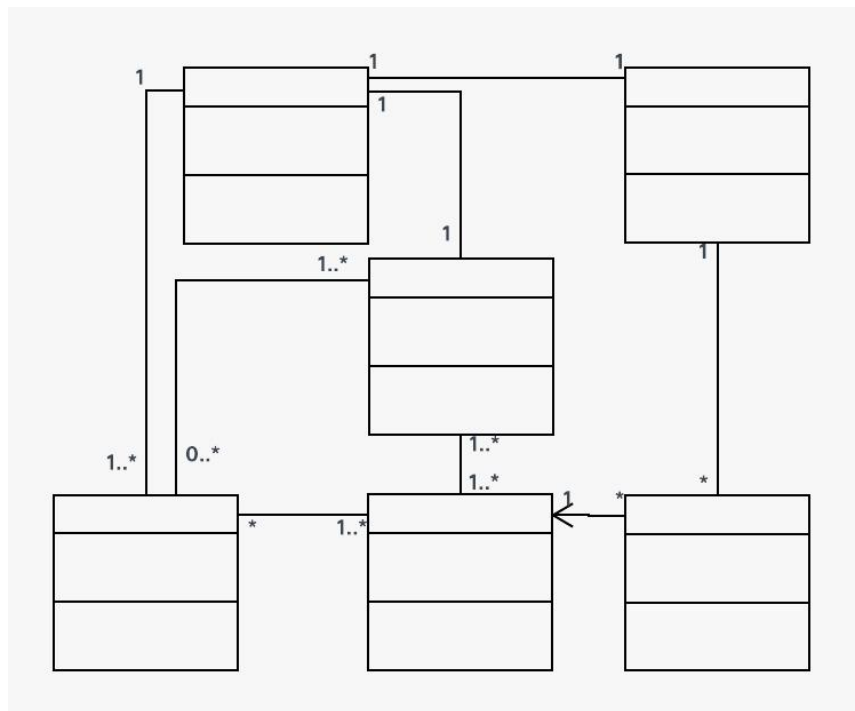


Gambar Use Case Diagram

2. Class Diagram

Class diagram merupakan diagram yang selalu ada di pemodelan sistem berorientasi obyek. *Class diagram*

menunjukkan hubungan antar class dalam sistem yang sedang dibangun dan bagaimana mereka saling berkolaborasi untuk mencapai tujuan. *Class diagram* digunakan untuk menggambarkan disain statis dari sistem yang sedang dibangun (Hermawan, 2004:27).

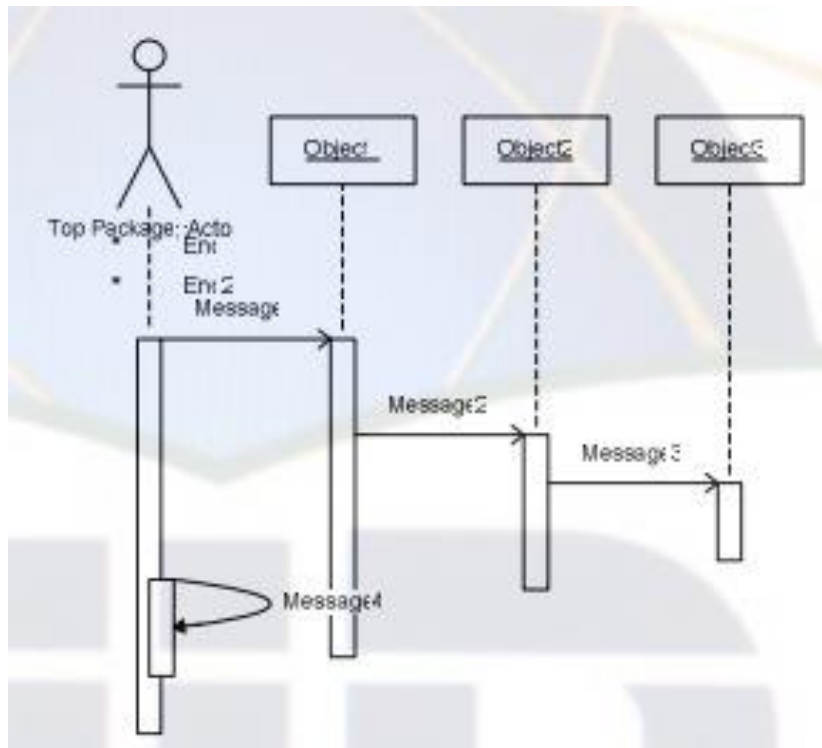


Gambar Class Diagram

3. *Sequence diagram*

Sequence diagram menjelaskan secara detail uruta proses yang dilakukan dalam sistem untuk mencapai tujuan dari use case: interaksi yang terjadi antar *class*, operasi apa saja yang terlibat, urutan antar operasi, dan informasi yang diperlukan oleh masing-masing operasi. *Sequence diagram* menjelaskan

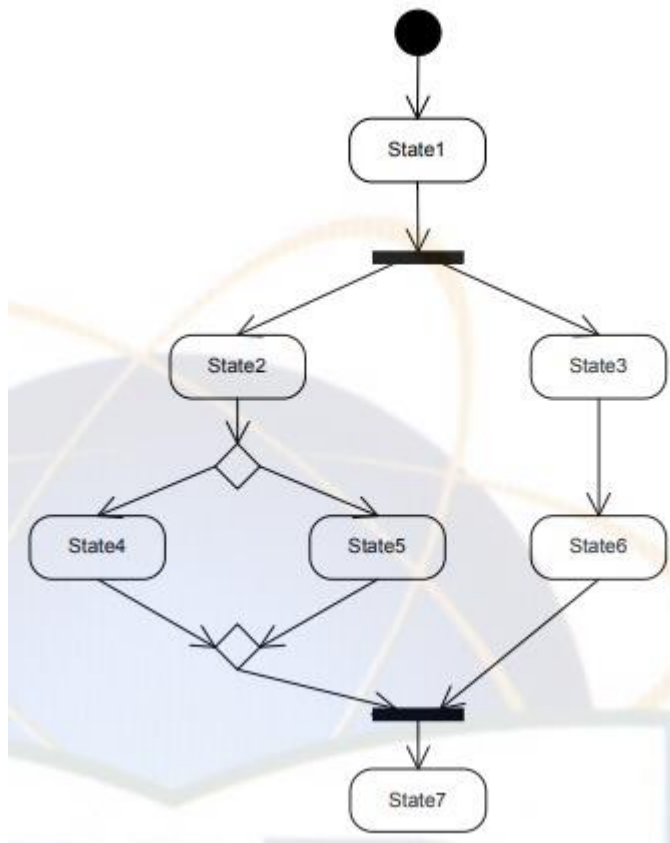
aspek dinamis dari sistem yang sedang dibangun (Hermawan, 2004:24).



Gambar Sequence Diagram

4. Activity Diagram

Activity diagram adalah teknik untuk mendeskripsikan logika prosedural, proses bisnis dan aliran kerja dalam banyak kasus. *Activity diagram* mempunyai peran seperti halnya *flowchart*, akan tetapi perbedaannya dengan *flowchart* adalah *activity diagram* mendukung perilaku paralel sedangkan *flowchart* tidak bisa (Munawar, 2005:109).

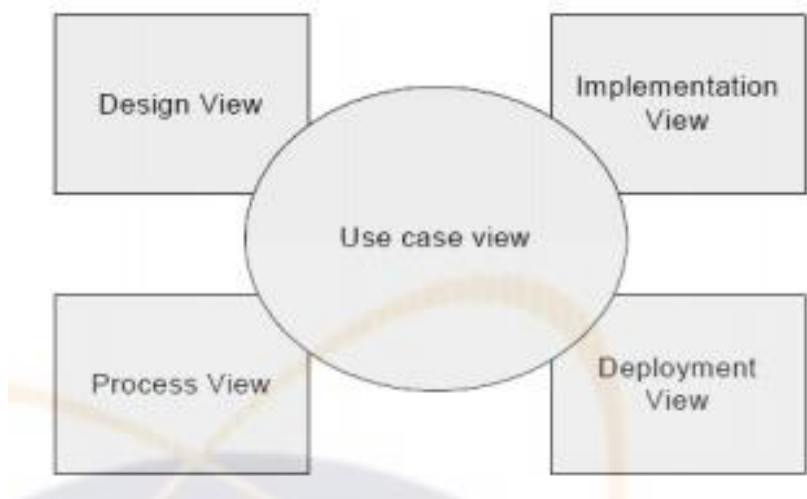


Gambar Activity Diagram

5. UML

UML dibangun atas model 4+1 view. Model ini didasarkan pada fakta bahwa struktur sebuah sistem dideskripsikan dalam 5 view dimana salah satu diantaranya *use case view*. *Use case view* ini memegang peran khusus diantaranya mengintegrasikan content ke *view* yang lain. Kelima view tersebut tidak berhubungan dengan diagram yang dideskripsikan di UML. Setiap *view* berhubungan dengan perspektif tertentu dimana sistem akan diuji. *View* yang

berbeda akan menekankan pada aspek yang berbeda dari sistem yang mewakili ketertarikan sekelompok *stakeholder* tertentu.



Gambar UML Diagram

Penjelasan lengkap gambar UML Diagram tentang sistem bisa dibentuk dengan menggabungkan informasi-informasi yang ada pada *view* pada kelima *view* tersebut.

1. *Use case view* mendefinisikan perilaku eksternal sistem.
2. *Design view* mendefinisikan struktur logika yang mendukung fungsi-fungsi yang dibutuhkan di *use case*. Informasi yang terkandung di *view* ini menjadi perhatian para programmer karena menjelaskan secara detail bagaimana fungsionalitas sistem akan diimplementasikan.

3. *Implementation view* menjelaskan komponen-komponen fisik dari sistem yang akan dibangun. Informasi yang ada di view ini relevan dengan aktifitas-aktifitas seperti manajemen konfigurasi dan integrasi sistem.
4. *Process sistem* berhubungan dengan hal-hal yang berkaitan dengan concurrency di dalam sistem.
5. *Deployment view* menjelaskan komponen-komponen fisik didistribusikan ke lingkungan fisik seperti jaringan komputer dimana sistem akan dijalankan.

2.12 Database

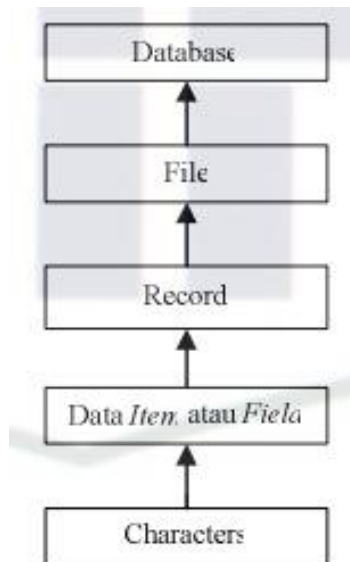


Gambar Database

Sumber : robicomp.com

Database didefinisikan sebagai kumpulan data yang saling berkaitan secara teknis, database juga sebagai tempat media

penyimpanan data kita dalam membuat sebuah program yang berisikan tabel, field, record, yang diselimuti dengan DBMS (*Database Management System*).[13]



Basis data (database) ini kumpulan dari data yang saling berhubungan satu dengan lainnya, tersimpan di perangkat keras komputer dan digunakan perangkat lunak untuk memanipulasinya (Jogiyanto, 2005:217). Jadi basis data merupakan suatu komponen utama sistem informasi karena semua informasi untuk pengambilan keputusan berasal dari data di basis data.

Sampai dengan membentuk suatu data base, data mempunyai jenjang yang dapat dilihat pada gambar (Jogiyanto, 1999:714-715):

a. Characters

Characters merupakan bagian data yang terkecil, dapat berupa karakter numerik, huruf atau pun karakter-karakter

husus (*special characters*) yang membentuk suatu item data atau field.

b. *Field*

Field menggambarkan suatu atribut dari record yang menunjukkan suatu item dari data, seperti misalnya nama, alamat dan lain sebagainya. Kumpulan dari field membentuk suatu record

1) Nama dari *field* (*field name*)

Field harus diberi nama untuk membedakan *field* yang satu dengan *field* yang lain.

2) Representasi dari *field* (*field representation*)

Representasi dari *field* menunjukkan tipe dari *field* (*field type*) dapat berupa tipe *numeric*, karakter atau huruf, tanggal, dan memo, serta lebar dari field (*field width*) menunjukkan ruang maksimum dari field yang dapat diisi dengan karakter-karakter data.

3) Nilai dari *field* (*field value*)

Nilai dari *field* menunjukkan isi dari *field* untuk masingmasing *record*.

c. *Record*

Record merupakan kumpulan dari *field* yang membentuk suatu *record*. *Record* menggambarkan suatu unit data individu tertentu. Kumpulan dari *record* membentuk suatu file. Misalnya file mahasiswa, tiap-tiap record dapat mewakili data tiap-tiap mahasiswa.

d. *File*

File terdiri dari *record-record* yang menggambarkan satu kesatuan data yang sejenis. Misalnya file mata kuliah berisi data tentang semua mata kuliah yang ada.

e. Database Database merupakan kumpulan dari file membentuk suatu database. Tujuan basis data yang efektif termuat di bawah ini:

- Memastikan bahwa data dapat dipakai diantara pemakai untuk berbagai aplikasi.
- Memelihara data baik keakuratan maupun konsistensinya.
- Memastikan bahwa semua data yang diperlukan untuk aplikasi sekarang dan yang akan datang akan disediakan secara cepat.
- Membolehkan basis data untuk berkembang dan kebutuhan pemakai untuk berkembang.
- Membolehkan pemakai untuk mengembangkan pandangan personalnya tentang data tanpa memperhatikan cara data disimpan secara fisik.

DataBase Management System (DBMS atau DMS) adalah paket perangkat lunak yang kompleks digunakan untuk memanipulasi database (Jogiyanto, 1999:731). Lebih detail lagi dijelaskan oleh Hariyanto bahwa DBMS adalah perangkat lunak untuk mendefinisikan, menciptakan, mengelola dan mengendalikan pengaksesan basisdata. Semua operasi *input* dan *output* yang berhubungan dengan database harus menggunakan DBMS. Bila pemakai akan mengakses database, DBMS menyediakan penghubung

(*interface*) antara pemakai dengan database (Jogiyanto, 1999:734).

Hubungan pemakai dengan database dapat dilakukan dengan cara:

- a. Secara interaktif menggunakan bahasa pertanyaan (*query language*).
- b. Dengan menggunakan program aplikasi.

Kegunaan Basis Data Penyusunan satu basis data digunakan untuk mengatasi masalah-masalah pada penyusunan data [2], yaitu:

- 1) Redudansi dan inkonsistensi data Jika file-file dan program aplikasi diciptakan oleh programmer yang berbeda pada waktu yang berselang cukup panjang, maka ada beberapa bagian data mengalami penggandaan pada file-file yang berbeda. Penyimpanan data yang berulang-ulang di beberapa file juga dapat mengakibatkan inkonsistensi (tidak konsisten).
- 2) Kesulitan Pengaksesan Data Suatu saat dibutuhkan untuk mencetak data siapa saja, padahal belum tersedia program yang telah tertulis untuk mengeluarkan data tersebut maka kesulitan tersebut timbul, dan penyelesaiannya untuk itu adalah kearah Sistem Manajemen Basis Data yang mengambil data secara langsung dengan bahasa yang familian dan mudah digunakan.
- 3) Isolasi data untuk standarisasi Jika data tersebar dalam beberapa file dalam bentuk format yang tidak sama, maka ini menyulitkan dalam menulis program aplikasi untuk mengambil dan menyimpan data, maka haruslah data dalam

satu basis data dibuat satu format sehingga mudah membuat program aplikasinya.

- 4) Masalah keamanan atau Security Setiap pemakai sistem basis data tidak semuanya diperbolehkan untuk mengakses semua data. Misalnya data mengenai gaji pegawai hanya boleh dibuka oleh bagian keuangan dan personalia. Keamanan ini dapat diatur lewat program yang dibuat oleh pemrogram atau fasilitas keamanan dari operating sistem.
- 5) Masalah Integrasi (Kesatuan) Basis Data berisi file yang saling berkaitan, masalah utama adalah bagaimana kaitan antara file tersebut terjadi. Meskipun diketahui bahwa file A berkaitan dengan file B, namun secara teknis maka ada file kunci yang mengaitkan kedua file tersebut.
- 6) Masalah data independence (kebebasan data) Aplikasi yang dibuat dengan bahasa yang diciptakan dari Sistem Manajemen Basis Data, apapun yang terjadi pada struktur file, setiap kali hendak melihat data cukuplah dengan utility USE, hendak menambah data cukup dengan APPEND, ini berarti perintah-perintah dalam paket Sistem Manajemen Basis Data bebas terhadap basis data. Perubahan apapun dalam basis data, semua perintah akan mengalami kestabilan tanpa perlu ada yang diubah.

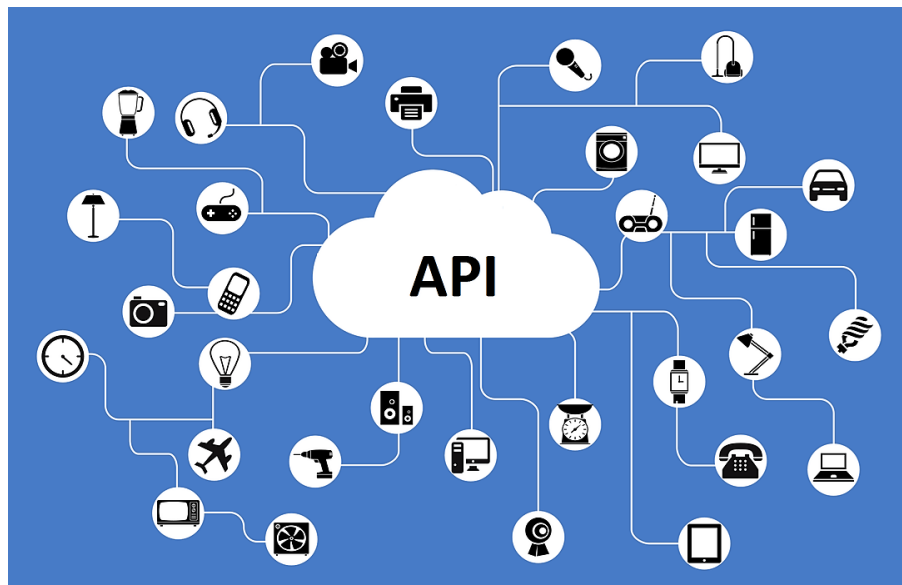
Berdasarkan dari uraian di atas dapat disimpulkan bahwa database adalah sekumpulan data yang beris tabel, field, dan record yang disimpan secara sistematis.

2.13 API (*Application Programing Interface*)

Application Programming Interface atau API adalah sebuah dokumentasi yang terdiri dari *interface*, kelas, fungsi, struktur dan sebagainya agar dapat membangun sebuah perangkat lunak. Dan API bisa dikatakan sebagai suatu kode pemrograman penghubung antara aplikasi atau web yang telah kita buat dengan fungsi yang dikerjakan [10]. Google+ api merupakan pemrograman yang menghubungkan ke Google+ [11].

Artinya dengan API, web PenA (*Penilaian Absensi*) akan terintegrasi dengan Rinfo+. Yang akan menghubungkan foto profile pada Rinfo+ dalam sistem PenA (*Penilaian Absensi*) . Dalam hal ini Google Api sebagai kode pemrograman yang disederhanakan yang dapat ditambahkan oleh sistem PenA (*Penilaian Absensi*) untuk mengakses dan terintegrasi dengan fitur Rinfo+.

Application Programming Interface (API) atau Antarmuka Pemrograman Aplikasi adalah sekumpulan perintah, fungsi, dan protokol yang dapat digunakan oleh programmer saat membangun perangkat lunak untuk sistem operasi tertentu. [14]



Gambar Application Programming Interface (API)

Sumber : hackernoon.com

Application Programming Interface (API) ini juga sekumpulan fungsi, perintah dan protokol yang dapat digunakan untuk menghubungkan satu aplikasi dengan aplikasi yang lain agar dapat berinteraksi. Seiring dengan perkembangan internet, API dapat diimplementasikan pada sisi *server* dan dapat digunakan oleh beberapa aplikasi yang dapat terhubung ke server dengan menggunakan protokol tertentu. Pada protokol HTTP, *Application Programming Interface* umumnya disebut sebagai *Web Application Programming Interface Server* atau *Web Service*.

Dengan adanya API ini, maka memudahkan programmer untuk “membongkar” suatu software, kemudian dapat dikembangkan atau diintegrasikan dengan perangkat lunak yang

lain. API dapat dikatakan sebagai penghubung suatu aplikasi dengan aplikasi lainnya yang memungkinkan programmer menggunakan sistem function. Proses ini dikelola melalui sistem operasi. Keunggulan dari API ini adalah memungkinkan suatu aplikasi dengan aplikasi lainnya dapat saling berhubungan dan berinteraksi. [15]

Application Programming Interface (API) ini yang memfasilitasi pertukaran informasi atau data antara dua atau lebih aplikasi perangkat lunak. API adalah antarmuka virtual antara dua fungsi perangkat lunak yang saling bekerja sama, seperti antara sebuah word processor dan sebuah spreadsheet [7]. Sebuah API mendefinisikan bagaimana cara programmer memanfaatkan suatu fitur tertentu dari sebuah komputer. API tersedia untuk sistem windowing, sistem file, sistem database, serta sistem jaringan.

Sistem informasi berbasis API memungkinkan sebuah back-end dimanfaatkan dengan cara yang lebih luas karena logic pada sistem dengan logic pada antarmukanya terpisah. Sistem informasi berbasis API juga akan mempermudah sebuah sistem untuk berkolaborasi dengan sistem lain. Hal ini jadi sangat berguna jika pada waktu mendatang proses bisnis yang difasilitasi menuntut interaksi dengan sistem lain, atau minimal akan sangat berguna pada pengembangan modul-modul baru pada sistem informasi terkait.

Secara umum API merupakan ekspresi terfokus keseluruhan fungsional dalam suatu modul software yang dapat diakses oleh orang yang membutuhkan dengan cara yang telah ditentukan layanan.

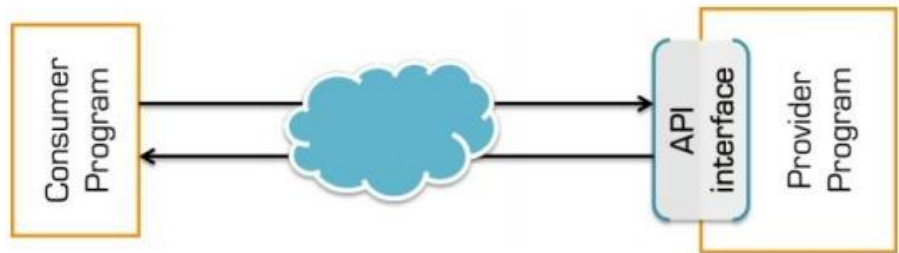
Representasi terfokus dari fungsi yang dideklarasikan dalam API dimaksudkan untuk menyediakan rangkaian layanan yang spesifik untuk target tertentu. Jika dalam satu modul memiliki API ganda, hal ini sudah menjadi hal yang umum karena setiap API dimaksudkan untuk penggunaan yang spesifik dari modul terkait (Rama dan Avinash, 2015).

Web API adalah antar muka program dari sistem yang dapat diakses melalui *method* dan *header* pada protokol HTTP yang standar. Web API dapat diakses dari berbagai macam HTTP client seperti browser dan perangkat mobile. Web API juga memiliki keuntungan karena menggunakan infrastruktur yang juga digunakan oleh web terutama untuk penggunaan *caching* dan *concurrency* (Miller dkk, 2014).

Windows Application Programming Interface (API) terdapat dalam file-file DLL yang menyusun Sistem Operasi Windows[14]. Karena terdapat dalam file *library* (DLL) maka fungsi-fungsi dan konstanta-konstanta tersebut dapat digunakan oleh aplikasi atau library lainnya. Windows API biasanya disusun dalam bahasa pemrograman C/C++. Bahasa C dipilih karena kecepatan dan kedekatannya dengan hardware.

Windows menyediakan banyak sekali API yang memiliki fungsi berbedabeda. Berikut ini adalah beberapa API yang disediakan oleh Sistem Operasi Windows :

1. Win32 API, API yang disediakan Windows untuk melakukan pemrograman Windows secara umum. Fungsi-fungsi yang terdapat dalam Win32 API ini sangat lengkap mulai dari menu, button, window, eksekusi program dan lain-lain.
2. *Telephony* API (TAPI), API yang digunakan untuk melakukan fungsifungsi telephony dengan menggunakan Sistem Operasi Windows.
3. *Messaging* API (MAPI), API yang digunakan untuk melakukan pengiriman pesan (termasuk pengiriman email) dengan menggunakan Sistem Operasi Windows.
4. *Speech* API (SAPI), API yang disediakan Windows untuk menambahkan fungsi speech ke dalam aplikasi yang berjalan di atas Sistem Operasi Windows.
5. *Cryptography* API, API untuk melakukan fungsi-fungsi enkripsi dan dekripsi. Banyak sekali metode enkripsi/dekripsi yang disediakan oleh cryptography API ini.
6. dan masih banyak API lainnya (termasuk yang tidak didokumentasikan oleh Microsoft). Seperti telah disinggung sebelumnya bahwa API sebenarnya hanyalah sebuah fungsi. Seperti halnya fungsi-fungsi lainnya, fungsi tersebut dapat memiliki parameter-parameter dan dapat mengembalikan nilai. Bedanya hanyalah fungsi-fungsi tersebut ditulis dalam bahasa C/C++ sehingga bagi yang belum familiar dengan bahasa C/C++ biasanya akan mengalami kesulitan.



Gambar 1 API

API memungkinkan programmer untuk menggunakan fungsi standar untuk berinteraksi dengan sistem operasi lain. API diimplementasikan dengan menulis fungsi panggilan atau sintaks dalam program, yang menyediakan sarana yang diperlukan untuk meminta layanan program.

Pada dasarnya, program API mendefinisikan cara yang tepat bagi pengembang untuk meminta layanan dari program itu. Sebagai contoh, Amazon.com merilis API sehingga pengembang situs web dapat lebih mudah mengakses informasi produk Amazon, menggunakan Amazon API.

Sebuah situs web pihak ke-tiga dapat memposting link langsung ke produk Amazon dengan harga yang terupdate dan pilihan untuk “beli sekarang”. (Prasetyo, 2014) Dalam API terdapat fungsi-fungsi/perintah-perintah untuk menggantikan bahasa yang digunakan dalam *system calls* dengan bahasa yang lebih terstruktur dan mudah dimengerti oleh pengembang. Fungsi yang dibuat dengan menggunakan API tersebut kemudian akan memanggil *system calls*

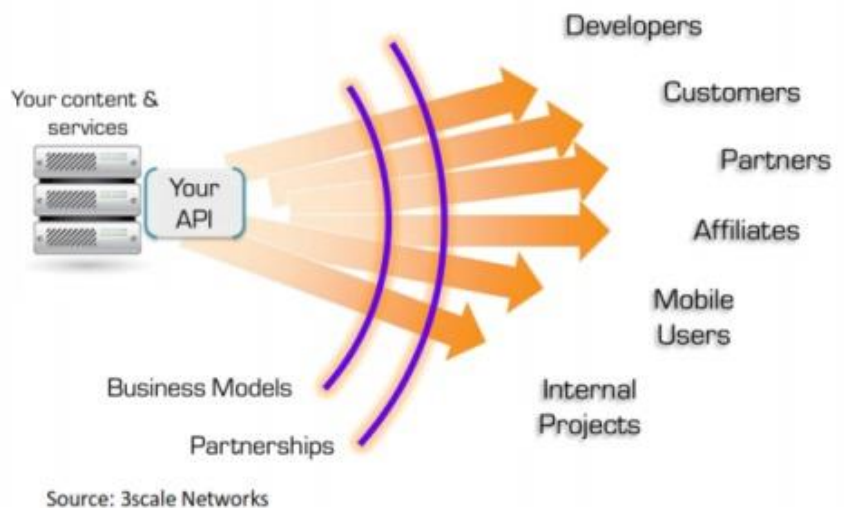
sesuai dengan sistem operasinya. Tidak tertutup kemungkinan nama dari systemcalls sama dengan nama di API.

Keuntungan memprogram dengan menggunakan API adalah:

1. Portabilitas

Pengembang yang menggunakan API dapat menjalankan programnya dalam sistem operasi mana saja asalkan sudah ter-install API tersebut. Sedangkan system call berbeda antar sistem operasi, dengan catatan dalam implementasinya mungkin saja berbeda (Gambar 1).

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 (Gambar 2).



Gambar 2 API

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 user 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 kita gunakan (Setiawan, 2014).

2.14 *Web Application*

Dewasa ini *web application* dikenal sebagai aplikasi yang diakses *melalui web browser* dan melalui jaringan seperti Internet atau intranet. Kemampuan untuk memperbarui dan memelihara aplikasi web tanpa harus mendistribusikan dan menginstal perangkat lunak pada kemungkinan ribuan komputer klien merupakan keunggulan teknologi ini, selain juga untuk cross-platform compatibility. Termasuk aplikasi web common webmail, penjualan ritel online, online pelelangan, wiki dan banyak fungsi lainnya.

Pada jurnal yang ditulis oleh Xu, dkk, (2005), menitikberatkan pada efektifitas dan efisiensi sebuah testing terhadap aplikasi yang berbasis web application dengan membandingkan dua metode yaitu Semantic Label dan XML description technique. Lei Xu dan timnya mengembangkannya dengan melengkapi mekanisme feedback control pada pembangunan aplikasi agar lebih menyempurnakan kualitas sistem. Edinburgh (2005) membahas sebuah pendekatan pengujian pada web application. Dalam metode pendekatannya analisa aliran data akan dianggap sebagai *Function Level Testing*, *Function Cluster Level Testing*, *Object Level Testing* dan *Web Application Level Testing*, dari level terendah hingga level tertinggi.

2.15 XAMPP



Gambar XAMPP

Sumber : commons.wikimedia.org

XAMPP adalah sebuah *software web server apache* yang didalamnya sudah tersedia database server MySQL dan dapat mendukung pemrograman PHP. XAMPP merupakan software yang mudah digunakan, gratis dan mendukung instalasi di Linux dan Windows. Keuntungan lainnya adalah cuma menginstal satu kali sudah tersedia Apache Web Server, MySQL Database Server, PHP Support (PHP 4 dan PHP 5) dan beberapa module lainnya.

Perbedaan versi untuk Windows operating system sudah dalam bentuk instalasi grafis dan yang Linux dalam bentuk file terkompresi tar.gz. Kelebihan lain yang berbeda dari versi untuk Windows adalah memiliki fitur untuk mengaktifkan sebuah server secara grafis, sedangkan Linux masih berupa perintah-perintah di dalam console.

2.16 *CSC (Cascading Style Sheets*



Gambar Logo CSS

Sumber : petanikode.com

CSS (*Cascading Style Sheets*) Pada versi HTML yang terdahulu, web browser mengontrol tampilan (rendering) dari setiap halaman web. Jika menggunakan elemen H1 pada (*large heading*) pada web dokumen, browser akan merender elemen tersebut.

Dengan adanya CSS, *programmer* dapat mengontrol bagaimana *browser me-render* halaman web. Mengaplikasikan CSS pada halaman web dapat memberikan tampilan yang lebih menarik dan spesifik sesuai dengan tema pada sebuah web site.

Teknologi CSS memberikan fasilitas untuk menentukan style (misal; *spacing, margins*) dari elemen halaman web terpisah dari struktur dokumen web (*section headers, body text, links*). Pemisahan tersebut memberikan peningkatan yang lebih besar dalam pengaturan *web pages*, dan membuat perubahan – perubahan *style* dalam dokumen dapat dilakukan lebih cepat dan lebih mudah.

2.17 *Local Host*

Localhost adalah sebuah akses local yang didapat dari sebuah aplikasi untuk dapat mengakses local server yang dimaksudkan untuk mengetahui bagaimana kekurangan dan kelebihan aplikasi yang akan digunakan sebelum dilakukan hosting.

Hosting adalah sebuah tempat dimana kita bisa menyimpan data-data website kita sehingga dapat diakses lewat

internet. Alasan peneliti menginstall di *localhost* sebelum *hosting* dikarenakan selain mengetahui kekurangan dan kelebihan aplikasi peneliti juga dapat menghemat penggunaan internet jika ada kesalahan yang dialami sebelum di *upload* pada *hosting* yang akan digunakan.

Localhost adalah server lokal yang ada pada komputer. Localhost adalah hostname dari komputer itu sendiri. Localhost dapat diakses ketika ada web server yang berjalan.

Local Host Menurut Kasiman Peranginangin (2006), *Localhost* adalah sebuah *host server* yang ada pada komputer Anda sendiri. *Localhost* biasanya digunakan untuk kebutuhan intranet atau untuk mempraktikkan pembuatan *web* sebelum di-*upload* ke *web hosting*. Salah satu penyedia server local adalah Xampp.

Xampp sendiri merupakan kepanjangan dari X, Apache, MySQL, PHP, dan Perl. X di sini maksudnya Xampp, yaitu cross (X) platform yang dapat digunakan pada sistem operasi Windows, Linux, Mac, dan Solaris. Sedangkan Apache ialah server yang digunakan. MySQL berarti basis data yang digunakan untuk localhost. PHP dan Perl merupakan script/bahasa pemrogramannya.

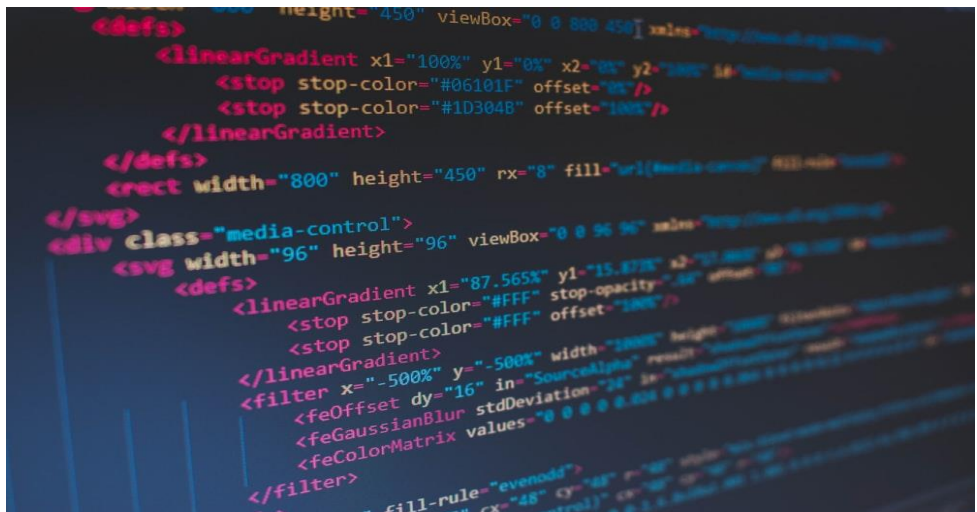
2.18 *Flow Map*

Flowmap adalah campuran peta dan *flow chart*, yang menunjukkan pergerakan benda dari satu lokasi ke lokasi lain, seperti jumlah orang dalam migrasi, jumlah barang yang diperdagangkan,

atau jumlah. *Flowmap* menolong analisis dan programmer untuk memecahkan masalah ke dalam segmen-segmen yang lebih kecil dan menolong dalam menganalisis alternatif-alternatif lain dalam pengoperasian.[5]

Flowmap merupakan diagram alir dokumen yang digunakan untuk menggambarkan hubungan antara *entity* yang terlibat berupa aliran-aliran dokumen yang ada. Untuk menjalankan prosedur sistem, digunakan *flowmap* yang terbentuk dari analisis prosedur. Bagan alir dokumen tersebut juga disebut bagan alir formulir yang merupakan bagan alir yang menunjukkan arus dari laporan dan formulir termasuk tembusan – tembusannya.

2.19 HTML



Gambar Script HTML

Sumber : animamedia.com

HTTP merupakan protokol yang digunakan untuk mentransfer data antara *web server* ke *web browser*. Protokol ini mentransfer dokumen-dokumen web yang ditulis atau berformat HTML. Dikatakan markup karena HTML berfungsi untuk "mepercantik" file text biasa untuk ditampilkan pada program web browser. Hal ini dilakukan dengan menambahkan tag-tag (perintah khusus) pada file teks biasa tersebut.

Tag HTML biasanya berupa tag-tag yang berpasangan dan ditandai dengan simbol `<dan>`. Pasangan dari sebuah tag ditandai dengan tanda `' '`. misalnya awalnya perintah tagnya `<contoh>` maka diakhiri dengan `</contoh>`.

2.20 WWW (*World Wide Web*)

World wide web atau WWW merupakan suatu program yang ditemukan oleh tim Beerners-Lee pada tahun 1991. Pada awalnya mereka hanya ingin menemukan suatu cara untuk menyusun arsip-arsip risetnya. Maka dari itu, mereka mengembangkan suatu sistem untuk keperluan pribadi. Sistem tersebut merupakan program peranti lunak dengan nama *enquire*. Dari program itulah tim mereka berhasil menciptakan jaringan yang menautkan berbagai arsip sehingga memudahkan pencarian informasi hal inilah yang menjadi dasar dari sebuah perkembangan pesat yang dikenal sebagai WWW.

Tim Beerners-Lee membuat pengajuan untuk proyek pembuatan *hypertext* global sekitar tahun 1989, sehingga pada bulan oktober 1990, *waring wera wanua* sudah dapat dijalankan dalam

lingkungan CERN (Pusat penelitian fisika partake Eropa). WWW secara resmi digunakan secara luas pada jaringan internet sekitar musim panas tahun 1991.

2.18.1 Bagaimana WWW Bekerja

World wide web bekerja berdasarkan tiga mekanisme sebagai berikut:

- Informasi disimpan dalam dokumen yang sering kita sebut dengan halaman *web*.
- Halaman *web* adalah file-file yang disimpan dalam komputer. komputer tersebut biasa disebut dengan istilah *web server*.
- Komputer yang mengakses dari halaman *web* tersebut biasa disebut dengan *web clients*.
- *Web clients* menampilkan halaman *web* dengan menggunakan program yang biasa dikenal dengan nama *web browser* seperti *chrome*, *firefox* dan *internet explorer*.

2.18.2 Bagaimana browser menampilkan web

- Semuan halaman *web* mengandung instruksi tentang bagaimana dia akan ditampilkan.
- *Browser* menampilkan beberapa halaman tersebut dengan cara membaca intruksi ini.
- Intruksi yang paling umum biasa disebut dengan *tag-tah* HTML.

Untuk lebih detailnya, berikut ini adalah ilustrasi bagaimana *browser* bekerja:

Misalnya kita ingin membuka sebuah halaman dengan URL `http://contoh.org/belajar/php` .

Yang pertama, *browser* memecah bagian nama *server* dari *URL*nya (`contoh.org`) ke dalam alamat protocol internet menggunakan basis data terdistribusi yang biasa disebut dengan istilah *Domain Name System* (DNS). DNS kemudian mencari nama url tersebut kemudian memberikan alamat IPnya seperti misalnya `192.168.1.100`, *browser* kemudian meminta *resource* dengan jalan mengirimkan *HTTP request* kepada komputer dengan alamat IP tersebut. *Browser* membuat *request* tersebut melalui *port* tertentu supaya *http request* yang lain (misalnya pengiriman email) dapat tetap ditangani. *Protocol* *http* biasanya menggunakan port 80.

Komputer yang menerima *http request* tersebut ke *web server*. Jika *web server* dapat memenuhi *request* tersebut dia akan mengirimkan *http response* ke *browser* dan memberikan status sukses semacam `http://1.0.200` OKE.

Web browser akan mengurai kode *html*, kemudian menerjemahkannya serta menuliskan teks yang terkandung didalam halaman *web* ke *layer*.

Banyak halaman *web* juga mengandung konten lain seperti gambar, *script*, *css*, dan lain-lain. Lalu *browser* akan membuat

http request tambahan pada *web server* untuk hal tersebut. Ketika *browser* menerima konten tersebut, *browser* secara progresif akan *render* konten tersebut dan menampilkannya ke *layer*.

BAB III

TATA CARA INSTALASI DAN PENGGUNAAN *TOOLS* YANG DIGUNAKAN

3.1 Xampp

3.1.1 Cara Install Xampp

1. *Download* aplikasi Xampp terbaru , link : <https://www.apachefriends.org/download.html>. Pilih salah satu sesuai dengan yang anda butuhkan :



The screenshot shows the XAMPP for Windows download page. It features a table with three rows representing different versions: 7.2.26 / PHP 7.2.26, 7.3.13 / PHP 7.3.13, and 7.4.1 / PHP 7.4.1. Each row includes a 'What's Included?' link, 'md5' and 'sha1' checksum links, a 'Download (64 bit)' button (which is highlighted with a blue box in the original image), and a 'Size' column. Below the table, there are links for 'Requirements', 'Add-ons', and 'More Downloads'. A note at the bottom states: 'Windows XP or 2003 are not supported. You can download a compatible version of XAMPP for these platforms [here](#).'

Version	Checksum	Size
7.2.26 / PHP 7.2.26	What's Included? md5 sha1	145 Mb
7.3.13 / PHP 7.3.13	What's Included? md5 sha1	146 Mb
7.4.1 / PHP 7.4.1	What's Included? md5 sha1	148 Mb

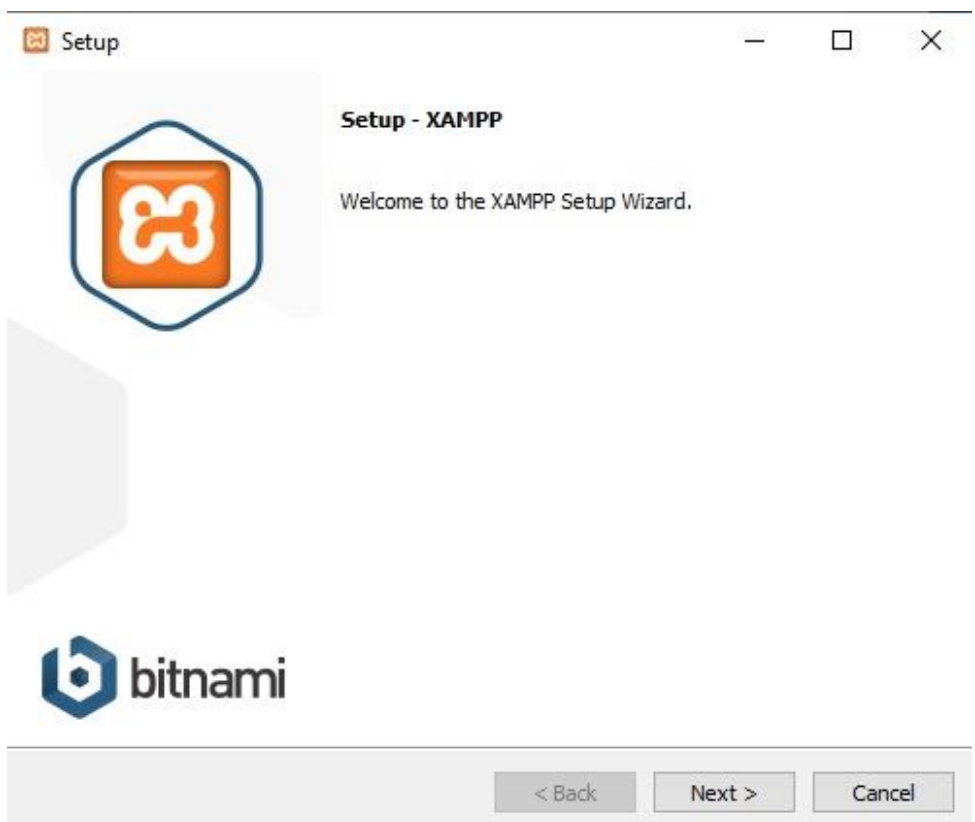
[Requirements](#) [Add-ons](#) [More Downloads »](#)

Windows XP or 2003 are not supported. You can download a compatible version of XAMPP for these platforms [here](#).

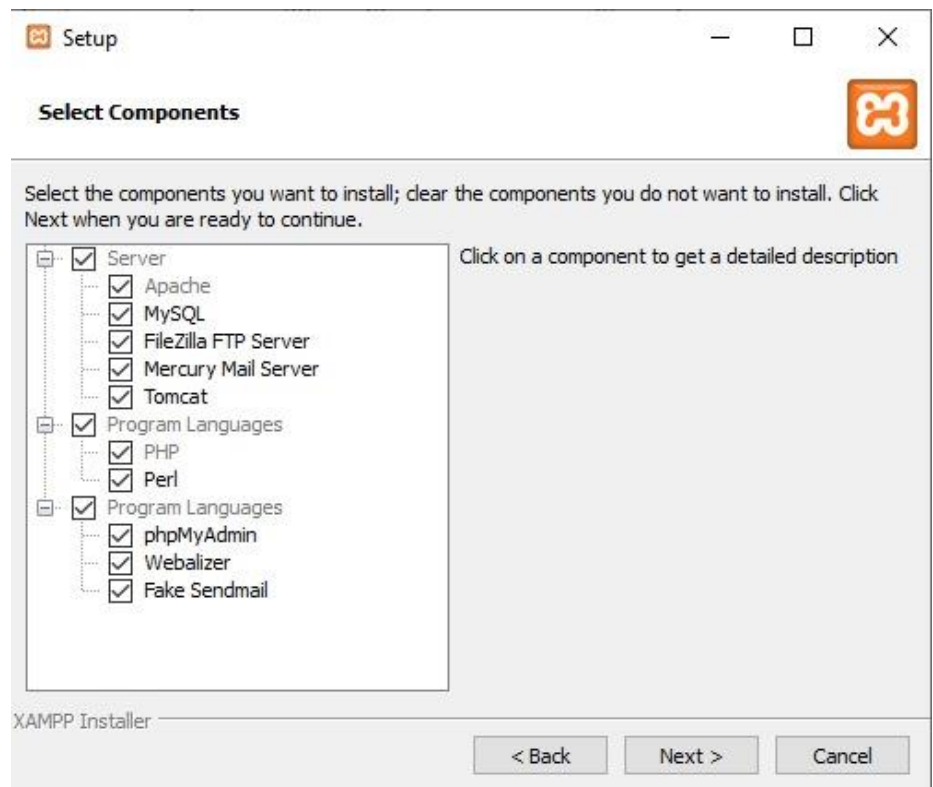
2. Klik dua kali pada file xampp yang baru saja anda *download*, jika muncul pesan *error* seperti gambar berikut abaikan saja dan pilih oke untuk melanjutkan instalasi



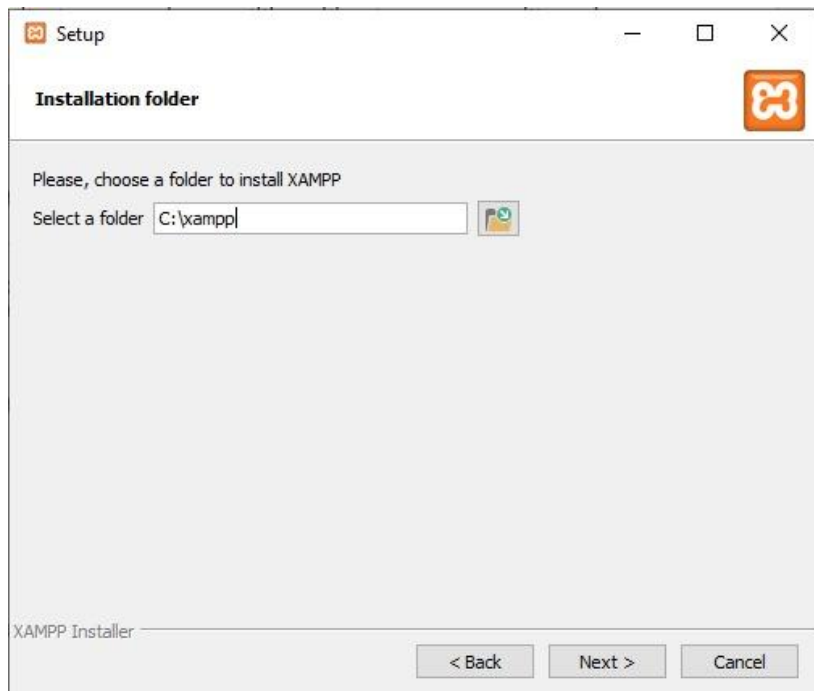
3. Selanjutnya anda akan dialihkan ke jendela yang isinya meminta anda untuk menginstall aplikasi Xampp



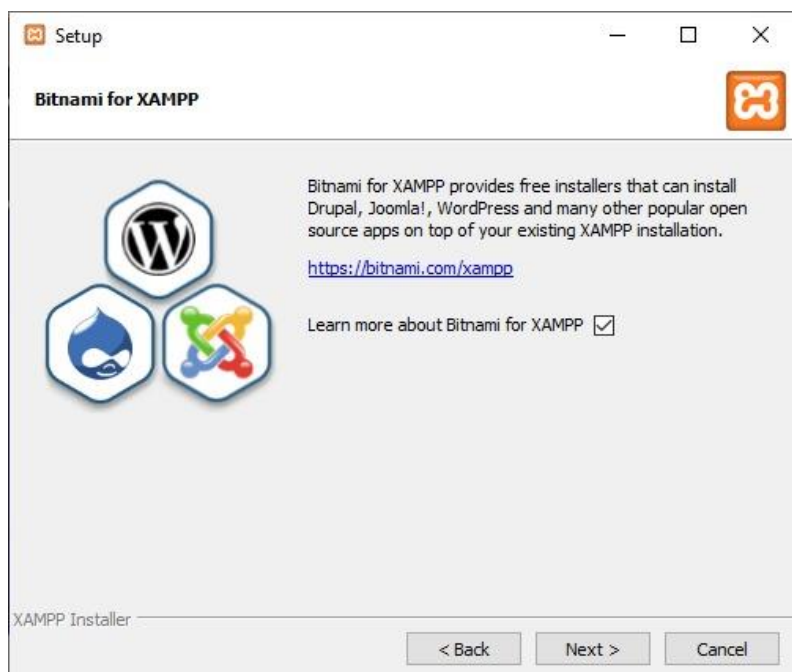
4. Klik *Next* untuk melanjutkan.
5. Selanjutnya anda akan diminta untuk memilih aplikasi yang mau di install. Centang saja semua pilihan dan klik tombol *Next*.



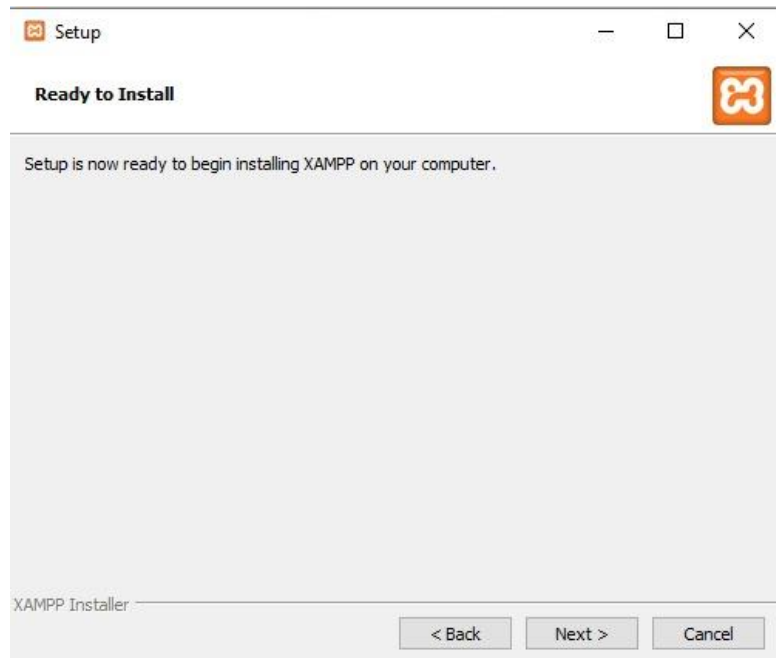
6. Kemudian anda akan diminta untuk menentukan lokasi folder penyimpanan file-file dan folder xampp. Secara default akan diarahkan ke lokasi C:\Xampp.
7. Tetapi jika anda ingin menyimpannya di folder lain anda bisa mengklik *browse* kemudian menentukan secara manual folder yang anda ingin gunakan.
8. Jika sudah selesai lanjutkan dan silahkan klik tombol install.



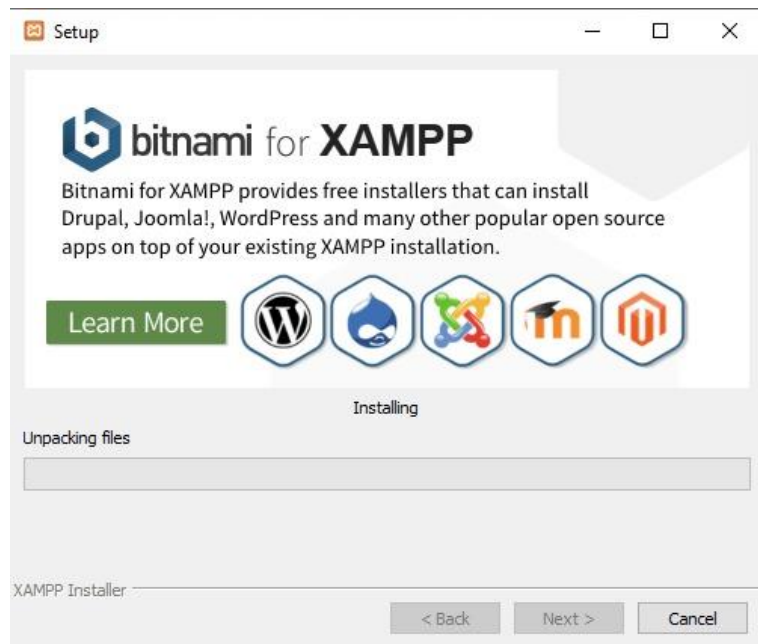
9. Selanjutnya akan muncul tampilan seperti gambar berikut, anda tinggal klik *Next* saja.



10. Selanjutnya akan muncul tampilan bahwa xampp siap untuk di install. Anda tinggal klik *Next* saja.

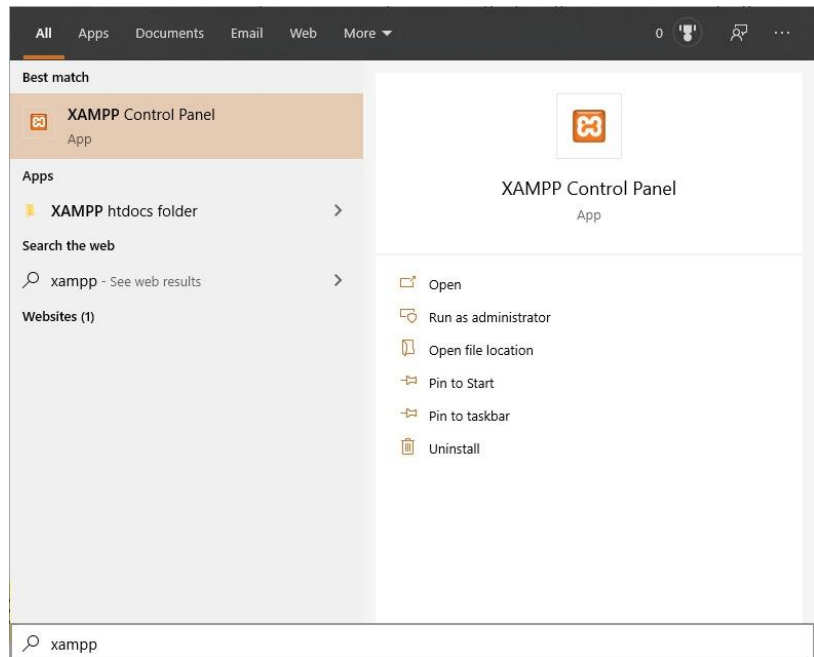


11. Tunggu sampai proses instalasi selesai.

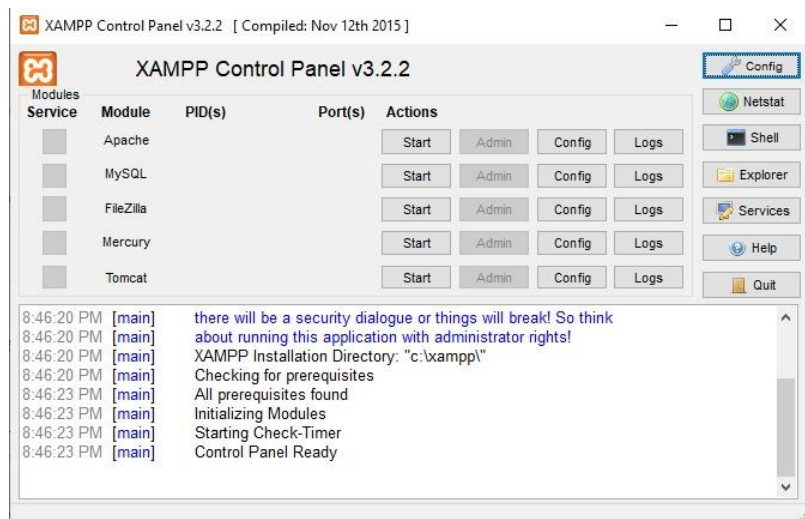


3.1.2 Cara Menjalankan Aplikasi Xampp

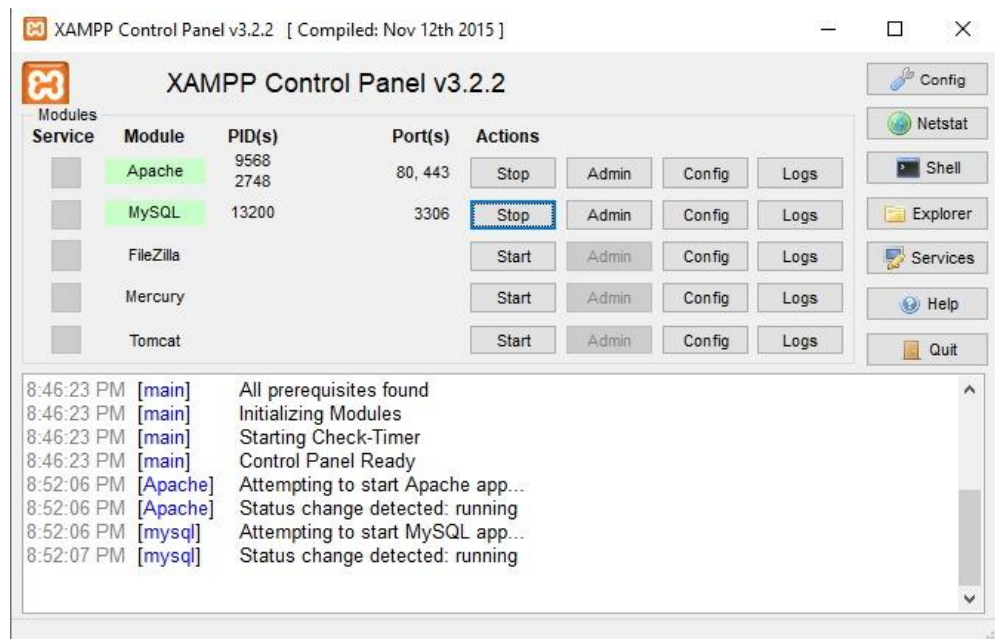
1. Bulakah aplikasi Xampp bisa melalui pencarian di task bar atau di desktop



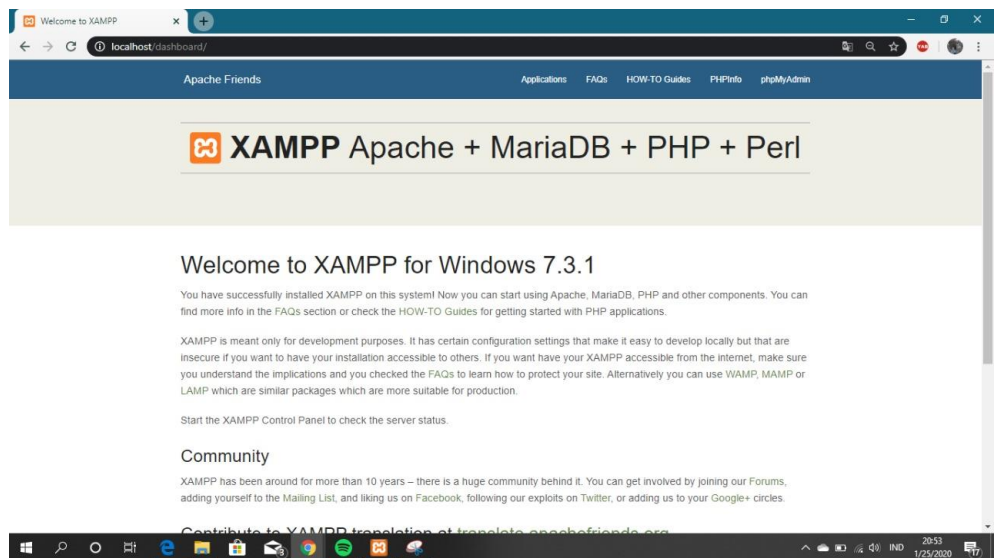
2. Klik dua kali pada xampp tersebut, maka akan muncul tampilan seperti berikut:



3. Silahkan klik tombol *Start* pada kolom *action* sehingga tombol tersebut berubah menjadi *stop*.
4. Dengan mengklik tombol tersebut artinya bahwa aplikasi tersebut telah dijalankan. Biasanya saya menggunakan xampp yang *Start* hanyalah aplikasi Apache dan MySQL karena saya tidak memerlukan aplikasi lainnya.



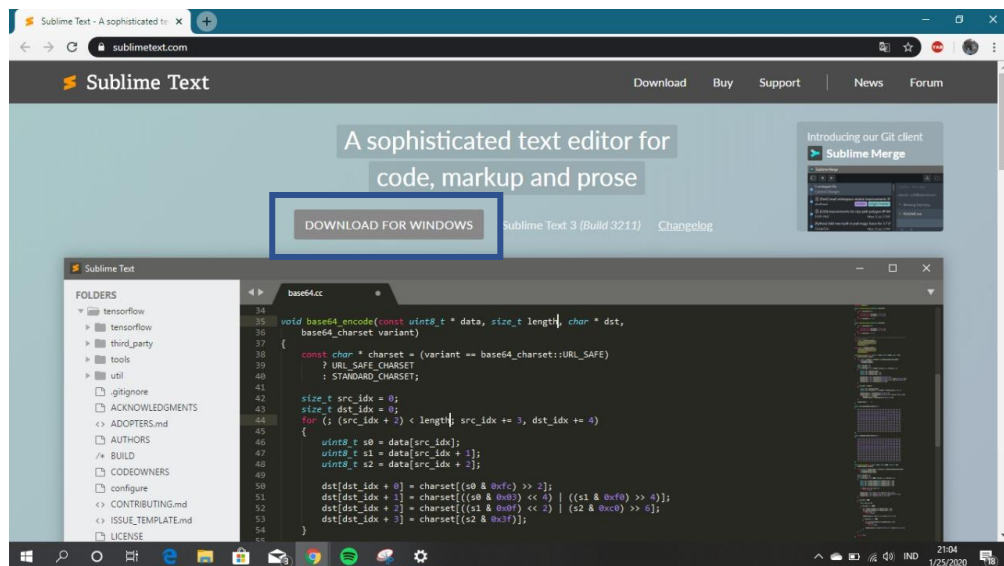
5. Selanjutnya buka *browser* anda dan coba ketikkan `http://localhost/` di *address bar*, jika muncul tampilan seperti gambar dibawah ini berarti proses instalasi telah berhasil.



3.2 Sublime

1. Download terlebih dahulu file sublime di situs resmi sublime,

link : <https://www.sublimetext.com/>



2. Setelah selesai di *download* klik dua kali pada file .exe

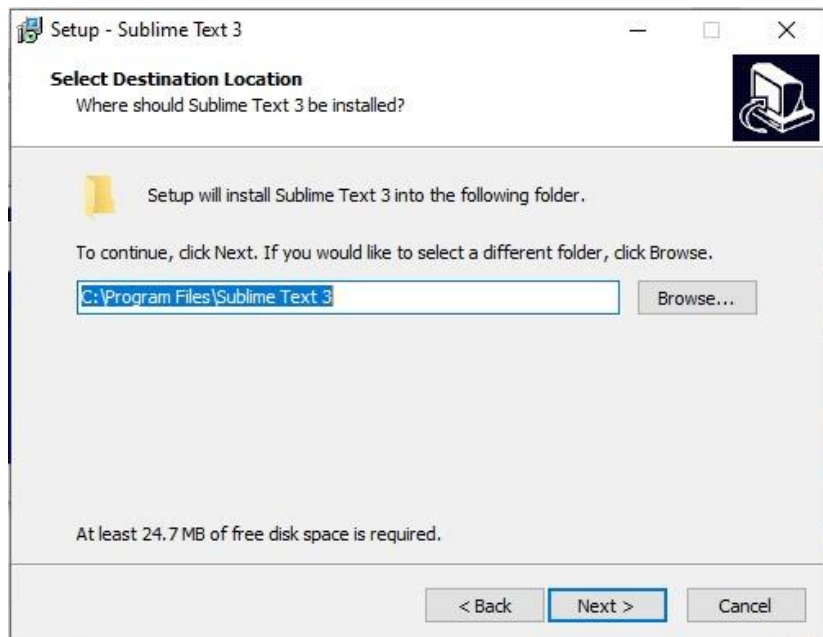


3. Selanjutnya akan muncul tampilan awal untuk proses instalasi.

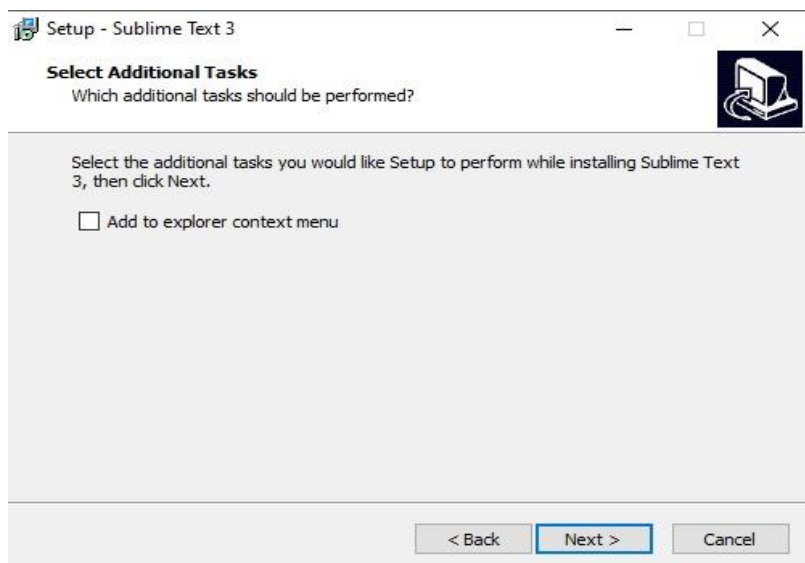


4. Klik *Next* untuk melanjutkan.
5. Setelah itu akan muncul tampilan library penyimpanan sublime. Secara default akan diarahkan ke lokasi C:\Program Files\Sublime Text 3

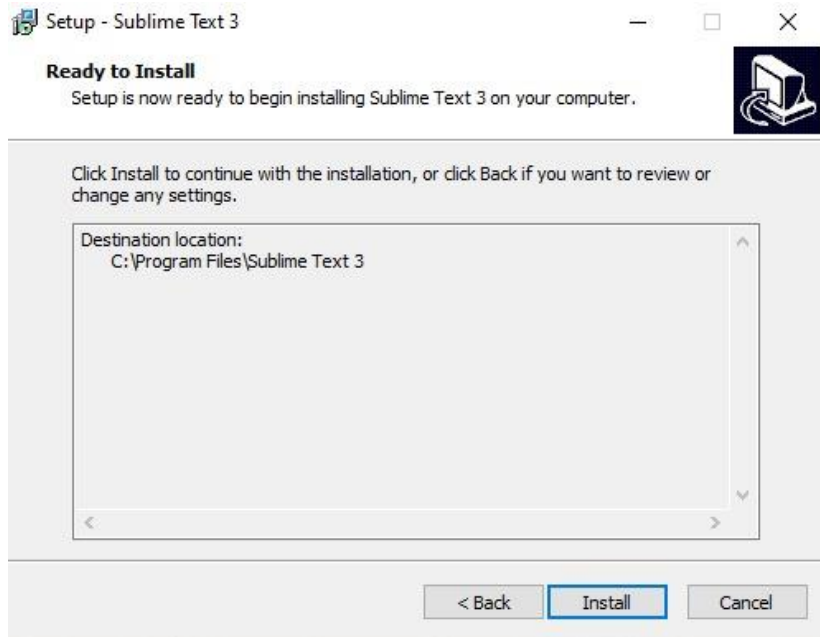
6. Tetapi jika anda ingin menyimpannya di folder lain anda bisa mengklik *browse* kemudian menentukan secara manual folder yang anda ingin gunakan.
7. Jika sudah selesai lanjutkan dan silahkan klik tombol *Next*.



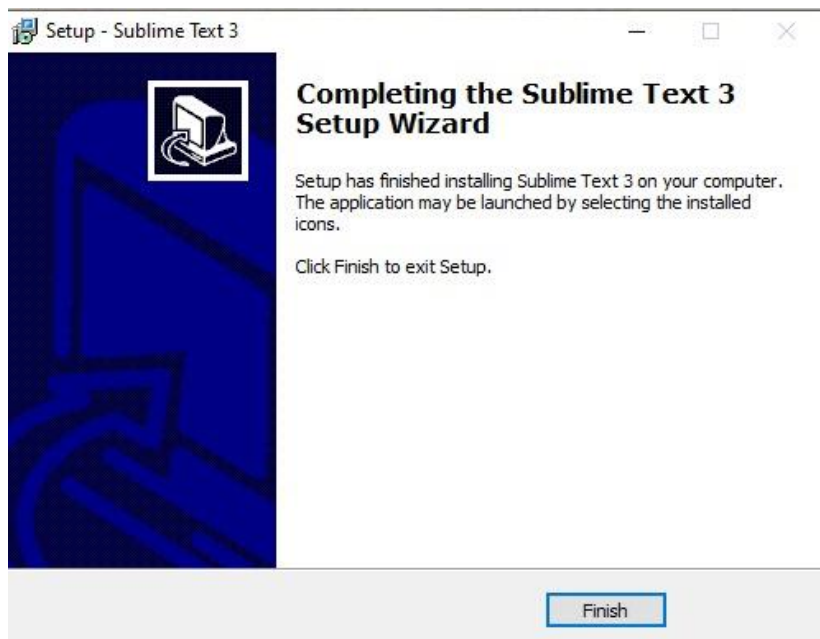
8. Selanjutnya akan muncul tampilan, abaikan dan klik *Next*.



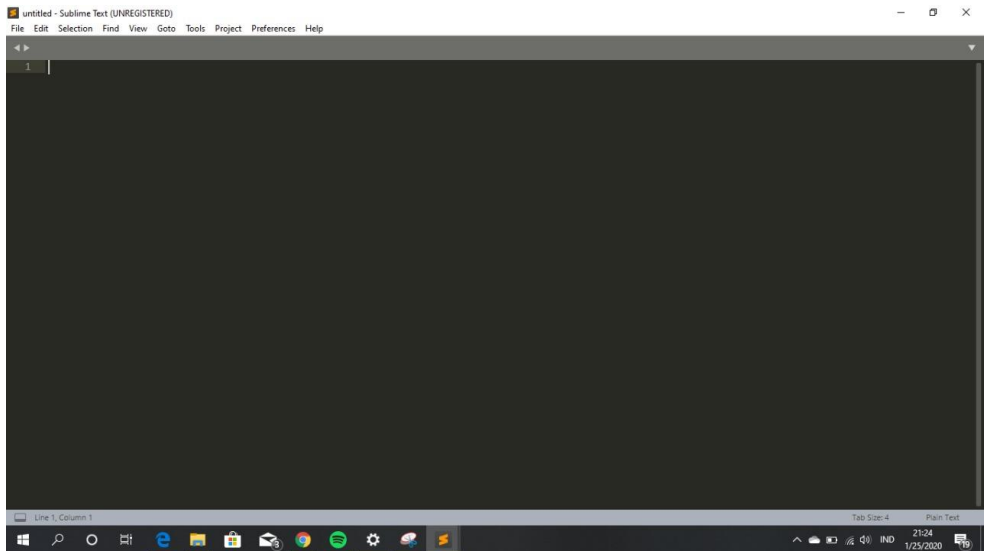
9. Setelah itu klik install untuk melanjutkan proses instalasi, tunggu sampai selesai.



10. Setelah proses instalasi berhasil klik *finish*.

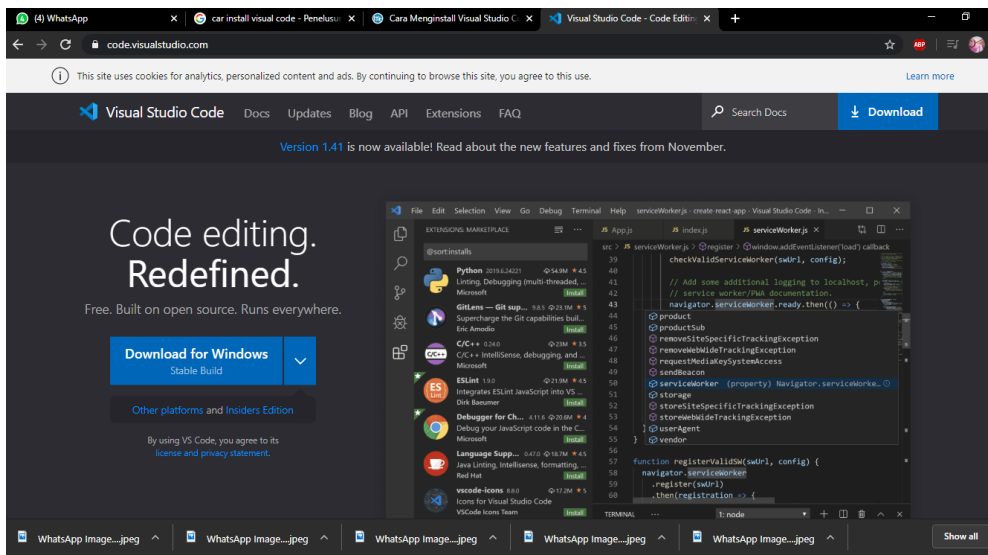


11. Tampilan awal sublime

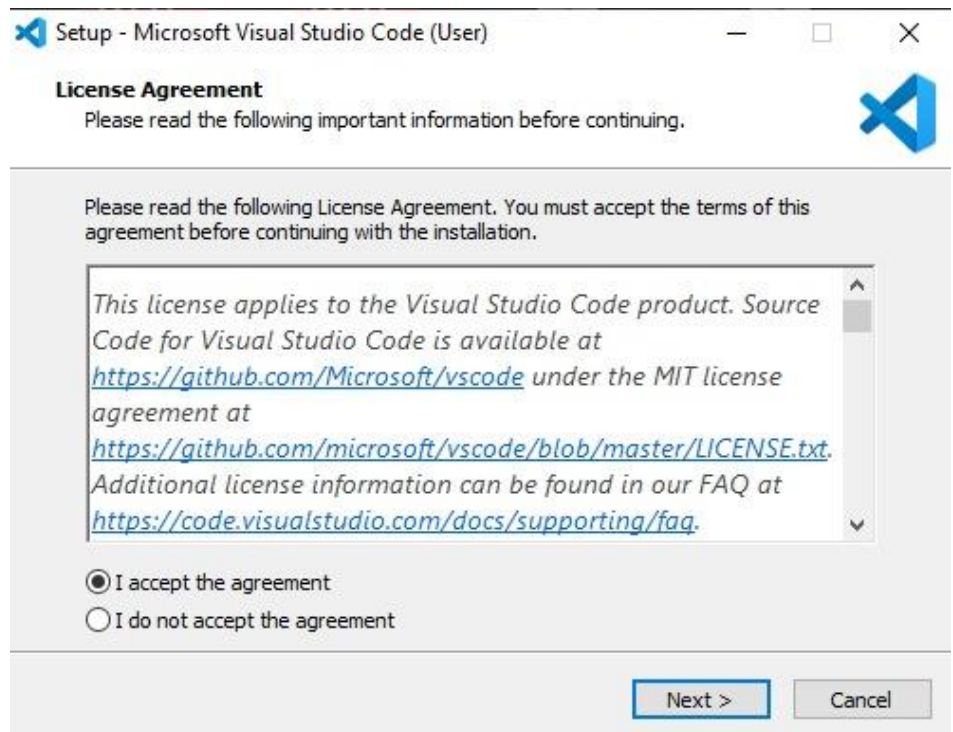


3.3 Visual Code

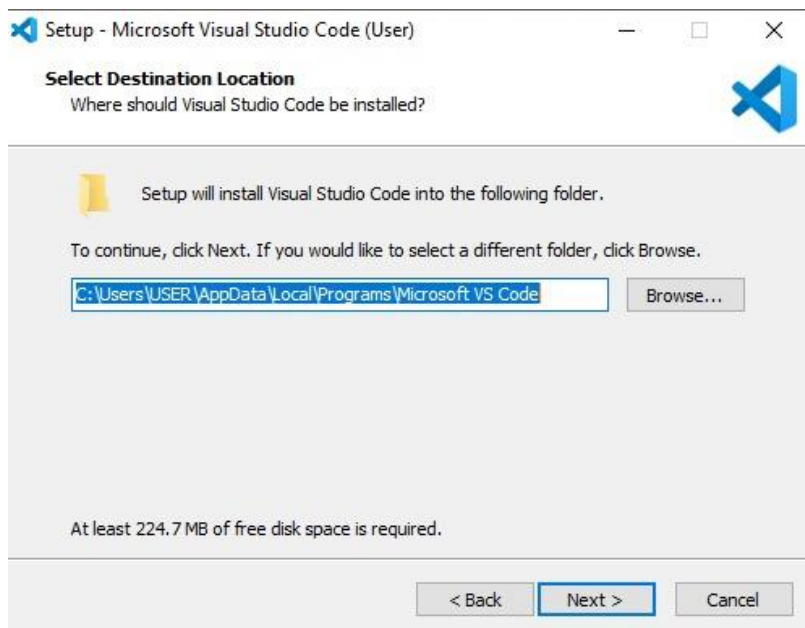
1. Pertama download aplikasi visual code, link <https://code.visualstudio.com/>.
2. Jika link sudah di buka klik download di pojok kanan . seperti pada gambar dibawah ini.



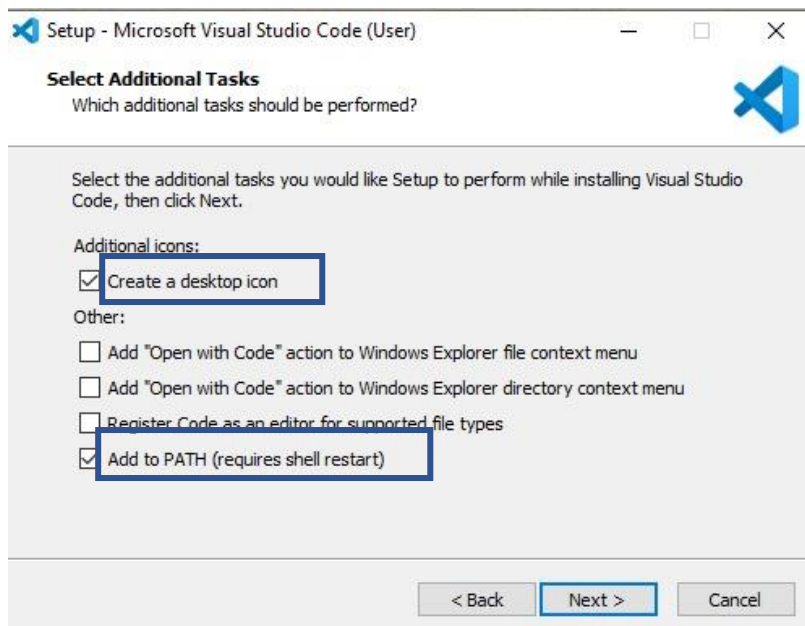
3. Setelah download selesai, buka aplikasi visual code yang telah didownload tadi, maka akan muncul *license agreement*.



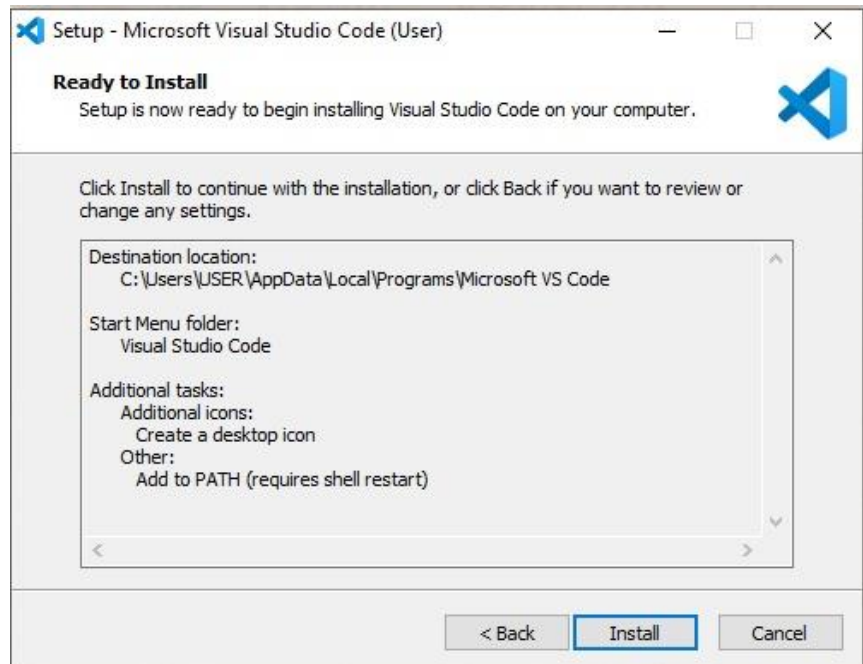
4. Pilih "*I accept the agreement*" untuk menyetujui kebijakan visual code, lalu untuk melanjutkan klik *Next*.
5. Setelah itu akan muncul tampilan library penyimpanan sublime. Secara default akan diarahkan ke lokasi C:\User\USER\AppData\Local\Programs\Microsoft VS Code
6. Tetapi jika anda ingin menyimpannya di folder lain anda bisa mengklik *browse* kemudian menentukan secara manual folder yang anda ingin gunakan.
7. Jika sudah selesai lanjutkan dan silahkan klik tombol *Next*.



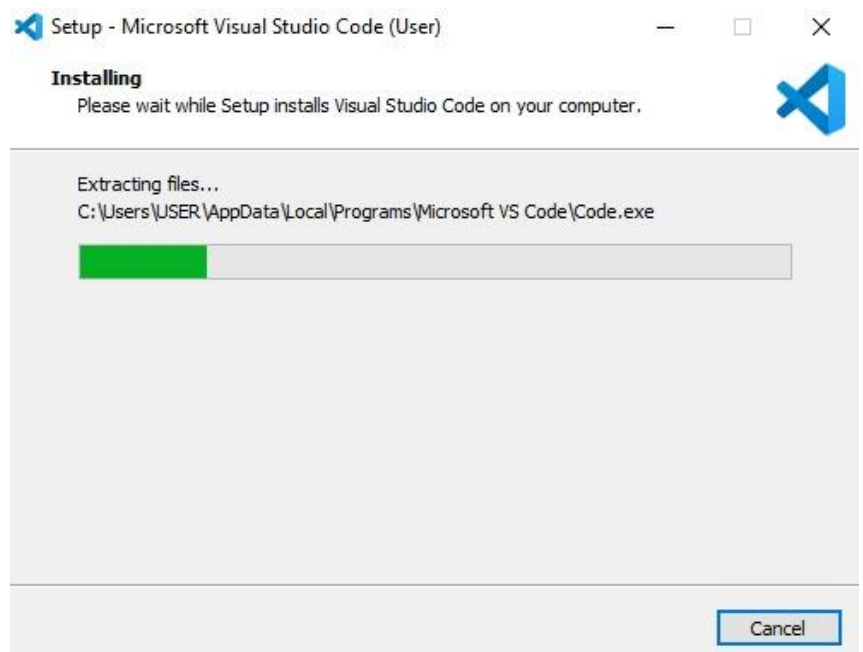
8. Setelah itu akan tampil seperti pada gambar di bawah ini, ceklis “*Create Dekstop Icon*”. Jika ingin membuat *shortcut* *VSCodenya*, ceklis “*Add to PATH (require shell restart)*”. Jika sudah klik *Next*.



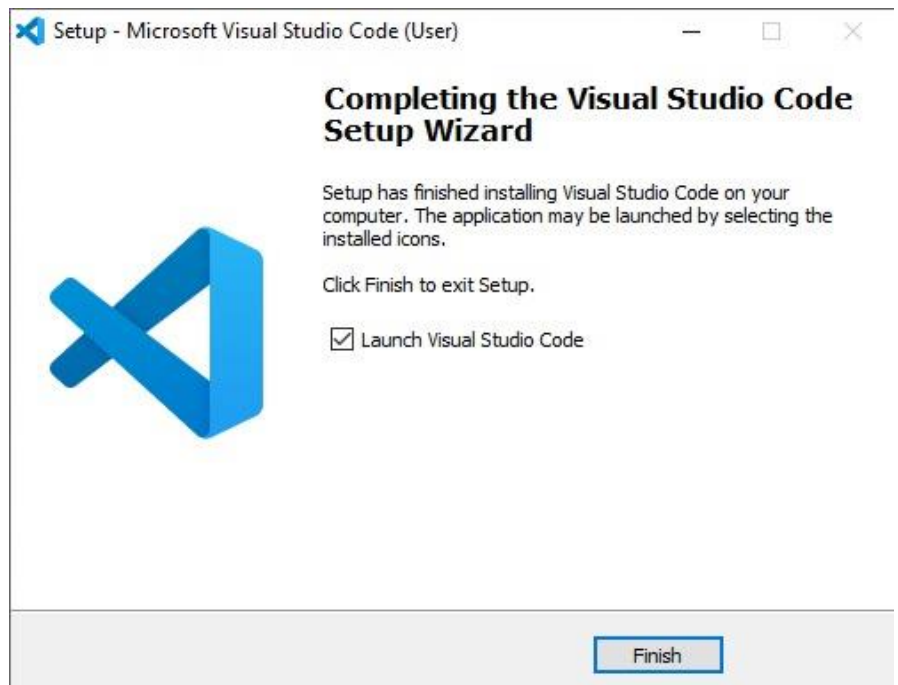
9. Selanjutnya klik *install* untuk melakukan proses instalasi.



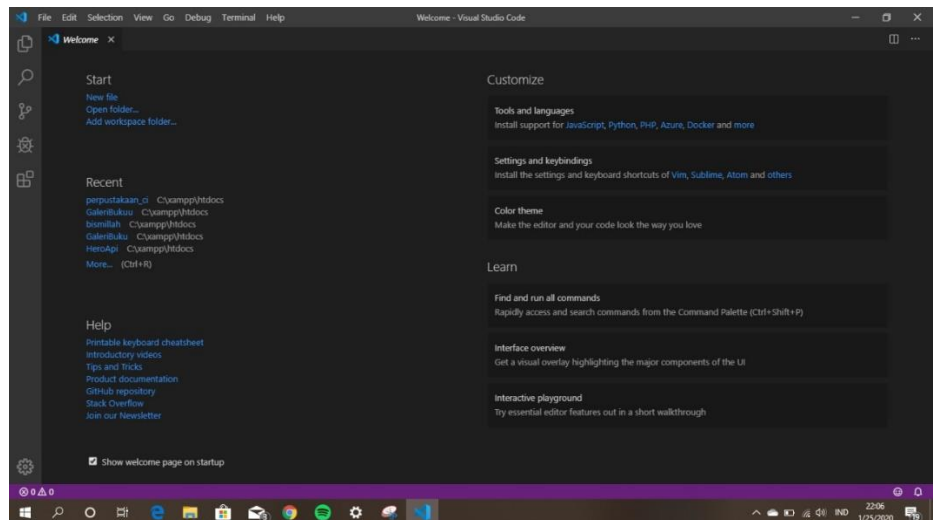
10. Proses instalasi dimulai, tunggu sampai proses selesai.



11. Setelah muncul tampilan seperti gambar dibawah ini, artinya instalasi telah berhasil.



12. Berikut adalah tampilan awal dari visual code.



BAB IV

PENJELASAN *FRAMEWORK* DAN BAHASA YANG DAPAT DIGUNAKAN

4.1 *Framework CodeIgniter*

4.1.1 Pengertian *Framework*

Framework adalah suatu kumpulan instruksi yang dikumpulkan dalam kelas dan *function-function* yang memiliki fungsi masing-masing untuk memudahkan *developer* dalam pemanggilan tanoa harus menuliskan *syntax* program yang sama berulang-ulang. Hal ini memiliki kegunaan untuk menghemat waktu dan mencegah penulisan *syntax* secara berulang-ulang agar terciptanya suatu *source code* yang bersih dan terstruktur.

Dalam *framework* sudah menyediakan fungsi *library* dan peralatan lainnya yang kita butuhkan. Salah satu *framework* yang cukup populer di Indonesia adalah *CodeIgniter*.

4.1.2 Pengertian *CodeIgniter*

CodeIgniter adalah sebuah *framework* PHP yang bersifat *open source* dan menerapkan salah satu metode MVC (*Model View Controller*). Dalam *CodeIgniter* anda bisa menggunakannya secara gratis atau bersifat *free* dengan kata lain tidak berbayar. *Framework CodeIgniter* dibuat dengan tujuan yang sama seperti *framework* lainnya yaitu; memudahkan *developer* atau programmer dalam membangun sebuah aplikasi berbasis web tanpa harus membuat semuanya dari awal.

CodeIgniter dirilis pertama kali pada tanggal 28 Februari 2006. *CodeIgniter* cocok digunakan untuk membuat suatu aplikasi web seperti :

- Portal berita,
- Sistem informasi,
- *Web start up*,
- *Profile company*,
- *E-Commerce*, dan
- *Blog*.

4.1.3 Kelebihan *CodeIgniter*

Ada beberapa kelebihan *codeigniter* dibandingkan dengan *framework-framework* yang lain yaitu :

- Performa cepat
Framework ini merupakan *framework* yang paling cepat dibandingkan dengan yang lain, karena tidak menggunakan *template engine* dan ORM yang dapat memperkambat suatu proses.
- Konfigurasi yang minim (*Nearly zero configuration*)
Untuk menyesuaikan dengan database dan keleluasaan *routine* tetap diizinkan melakukan konfigurasi dengan mengubah beberapa file seperti *database.php* atau *autoload.php*, namun dalam menggunakan CI hanya menerapkan aturan standar, kita hanya perlu mengubah sedikit saja pada file di folder *config*.
- Memiliki banyak komunitas
Di Indonesia komunitas CI lumayan ramai, selain itu tutorialnya pun mudah dicari dan kadang-kadang bisa dijadikan ajang diskusi dalam komunitas tersebut.

- Dokumentasi yang lengkap

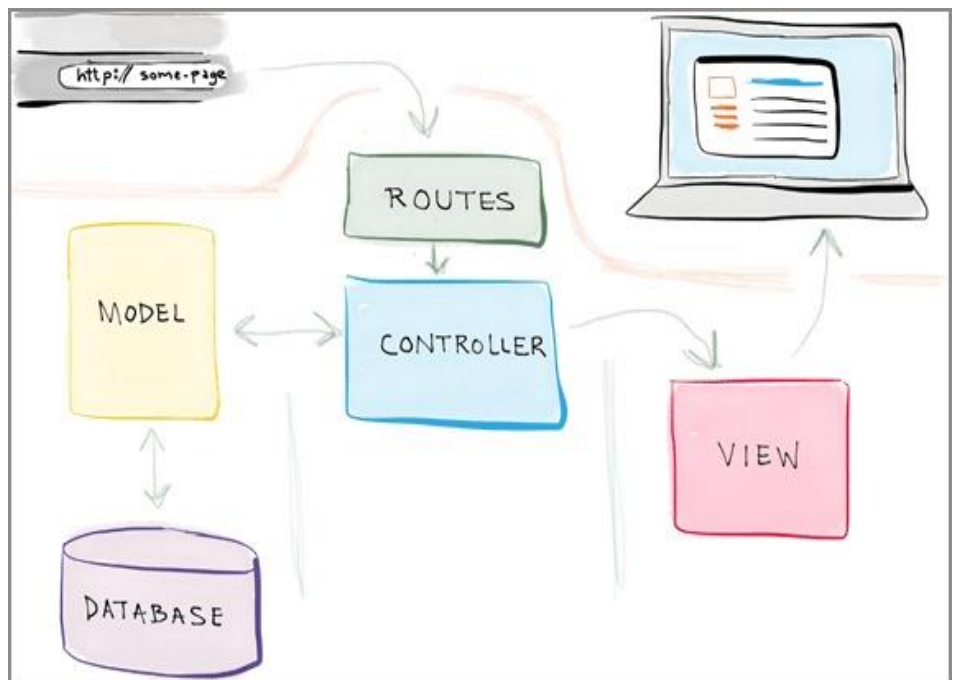
Ketika kita mendownload *CodeIgniter* didalam folder tersebut sudah disertai dengan *user_guide* yang berisi dokumentasi yang lengkap dan bisa diakses secara *offline*.

- Mudah dipelajari pemula

Bagi pemula *codeigniter* sangat mudah dipelajari karena tidak perlu bergantung pada *tools* tambahan seperti composer ORM *template engine*.

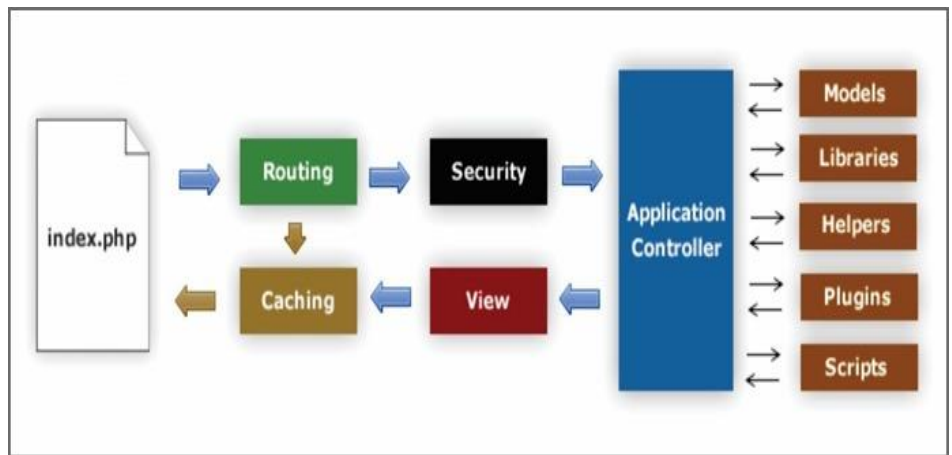
- Metode *Model View Controller*

Model view controller atau MVC adalah suatu Teknik atau konsep yang memisahkan komponen utama menjadi tiga komponen yaitu *model*, *view* dan *controller*.



- *Model, model* adalah suatu bagina penanganan yang berhubungan dengan pengelohan atau manipulasi database, misalnya mengambil data dari database, *meninput* dari pengelohan database lainnya, sehingga semua instruksi yang berhubungan dengan pengelolaan database diletakkan didalam model.
- *View, View* merupakan suatu bagian yang menangani halaman *user interface* atau halaman yang muncul pada *user*. Tampilan dari *user interface* dikumpulkan di *view* untuk memisah dengan *controller* dan *model* sehingga memudahkan *web designer* dalam melakukan pengembangan tampilan suatu halaman *website*.
- *Controller,controller* adalah suatu kumpulan instruksi atau aksi yang menghubungkan model dan *view* jadi *user* tidak akan berhubungan dengan model secara langsung begitu juga dengan *view* karena *controller* yang mengelolah instruksi.

Didalam *codeigniter* secara detail konsep MVC digambarkan sebagai berikut:



Pada gambar diatas file *index.php* memiliki peran sebagai *controller* utama yang memanggil fungs-fungsi dasar yang digunakan untuk menjalankan *controller*. *Router* memeriksa *HTTP request* selanjutnya memuttusakn *controller* mana yang akan digunakan untuk menangani *request* tersebut.

Apabila file *cache* tersedia alur aplikasi akan dilewati dan file *cache* tersebut akan dikirimkan ke *browser* pengguna. Sebelum *controller* dipanggil, *HTTP request* dan data yang dikirimkan pengguna akan di sortir terlebih dahulu dengan alasan keamanan. Kemudian *controller* memanggil file model, *library*, *helper* dan file lain yang dibutuhkan untuk menangani *HTTP request*. Hasil akhirnya akan ditampilkan oleh file *view* setelah itu dikirimkan ke *browser* untuk di tampilkan. Jika modus *chacing* diaktifkan hasil *view* akan di *chace* terlebih dahulu. Sehingga jika ada *request* yang sama bisa langsung bisa digunakan.

- Mendukung PHP4 dan PHP5

Saat ini PHP telah mencapai versi 5 bahkan versi tapi masih banyak juga orang-orang yang masih menggunakan PHP4,

oleh karena itu pengembang *framework codeigniter* memperhatikan betul setiap pengguna sehingga *framework codeigniter* dikembangkan agar mampu berjalan baik menggunakan PHP versi apapun

- *Codeigniter* mendukung banyak RDBMS (*Relational Database Management System*) seperti *MySQL*, *MySQLi*, *SQL Server*, *Oracle*, *Maria DB*, *PostgreSQL*, *SQLite*, dan lainnya.
- Mendukung SEO (*Search Engginie Optimization*) karena *codeigniter* pada dasarnya menganut *clean URL* sehingga setiap aplikasi yang dibangun menggunakan *codeigniter* lebih mudah di *index* oleh *search engine* populer seperti google, yahoo, msn dan lainnya.
- *Codeigniter* mengijinkan pengembangan web menggunakan *library* atau *helper* yang tidak disediakan oleh *codeigniter* seperti google map API, Facebook API, Fpdf dan lain sebagainya.
- *Codeigniter* bersifat tidak kaku sehingga memberika kebebasan kepada *developer web* untuk mengembangkan aplikasi berbasis *web* bahkan tanpa *framework*.
- Kompatibilitas dengan *Hosting*
Codeigniter mampu berjalan pada hamper semua *platform hosting* karena *codeigniter* mendukung semua database yang paling umum atau yang sering digunakan.
- Sangat Mudah di Integrasikan
Codeigniter sangat mengerti tentang pengembangan berbagai *library* saat ini. Maka dari itu *codeigniter* memberikan

kemudahan untuk diintegrasikan dengan *library-library* yang tersedia saat ini.

- *Small Footprint*

Small footprint yaitu minim jejak, atau lebih sederhananya ketika *user* melihat tampilan website *codeigniter* maka akan sulit ditebak. Hal ini sangat berguna bagi *developer*, ketika ada musuk *attacker* lebih sulit melihat *framework* yang digunakan contohnya saja penggunaan *wordpress* yang dapat langsung dilihat hanya menggunakan meta datanya.

4.1.4 Kekurangan CodeIgniter

Dalam *codeigniter* tidak hanya memiliki kelebihan saja sama seperti *framework* lainnya *codeigniter* juga memiliki kekurangan yaitu diantaranya:

- *Codeigniter* tidak ditujukan atau disarankan untuk pembuatan web dengan skala yang besar.
- *Library* yang sangat terbatas. Hal ini dikarenakan sangat sulit mencari *plugin* tambahan yang terverifikasi secara resmi, karena pada situsnya sendiri tidak menyediakan *plugin-plugin* tambahan untuk mendukung pengembangan aplikasi dengan CI.
- Belum memiliki editor khusus *codeigniter* sehingga dalam membuat suatu *project* dan modul-modulnya harus berpindah-pindah antara satu folder dengan folder yang lainnya.
- *Codeigniter* dikembangkan oleh Ellislab yang dimana bukanlah suatu komunitas sehingga menyebabkan *update core engine* nya tidak secepat *framework* yang lain.

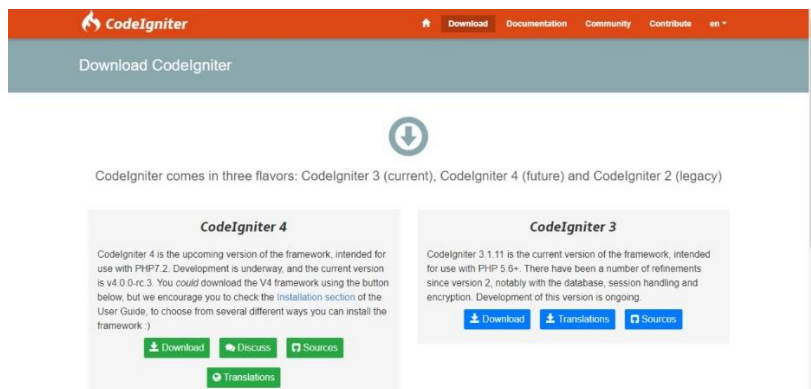
- Memiliki banyak kelonggaran dalam hal *coding* contohnya bebas menambahkan file.
- *Codeigniter* tidak mencerminkan MVC yang sesungguhnya, contohnya penulisan *echo* masih dapat dilakukan pada file *controller*.

Dari penjelasan MVC diatas dapat di simpulkan bahwa *controller* sebagai penghubung *view* dan model. Contohnya pada aplikasi yang menampilkan data dengan menggunakan metode MVC, *controller* memanggil instruksi pada model yang mengambil data pada database, kemudian *controller* yang meneruskannya pada *view* untuk ditampilkan.

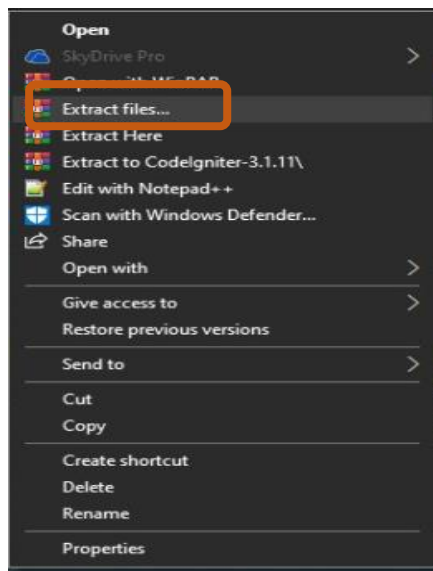
Jadi sudah jelas bahwa dalam pengembangan aplikasi dengan menggunakan konsep MVC ini sangat mudah dan membantu *web designer* atau *frontend developer* tidak perlu lagi berhubungan dengan *view* untuk mendesign tampilan aplikasi, sedangkan bagian *backend developer* mereka yang menangani bagian *controller* dan modelnya. Jadi pembagian tugas pun menjadi pun dan pengembangan aplikasi dapat dilakukan dengan cepat.

4.1.5 Cara Menggunakan CodeIgniter

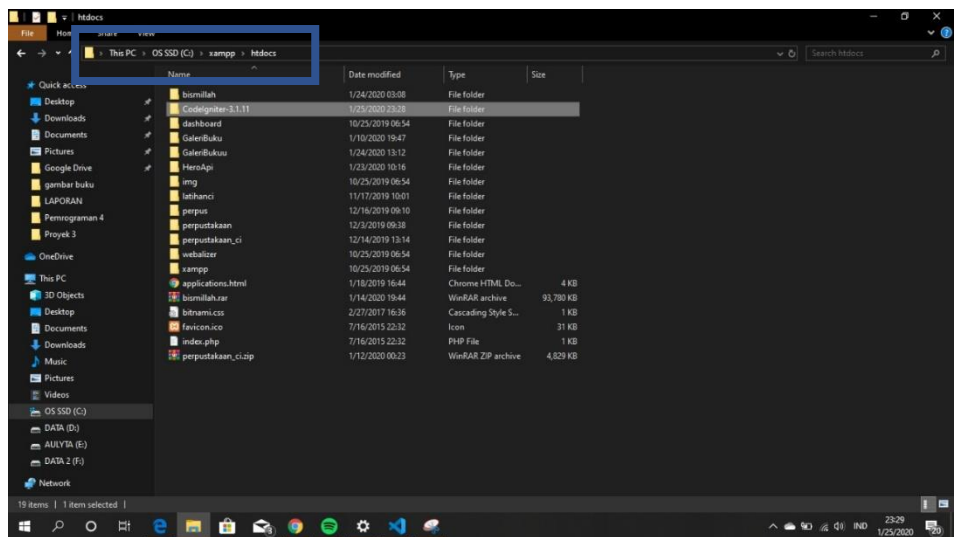
1. Download terlebih dahulu *codeigniter* di link <https://codeigniter.com/download>
2. setelah link tersebut di buka , pilih sesuai dengan kebutuhan masing masing



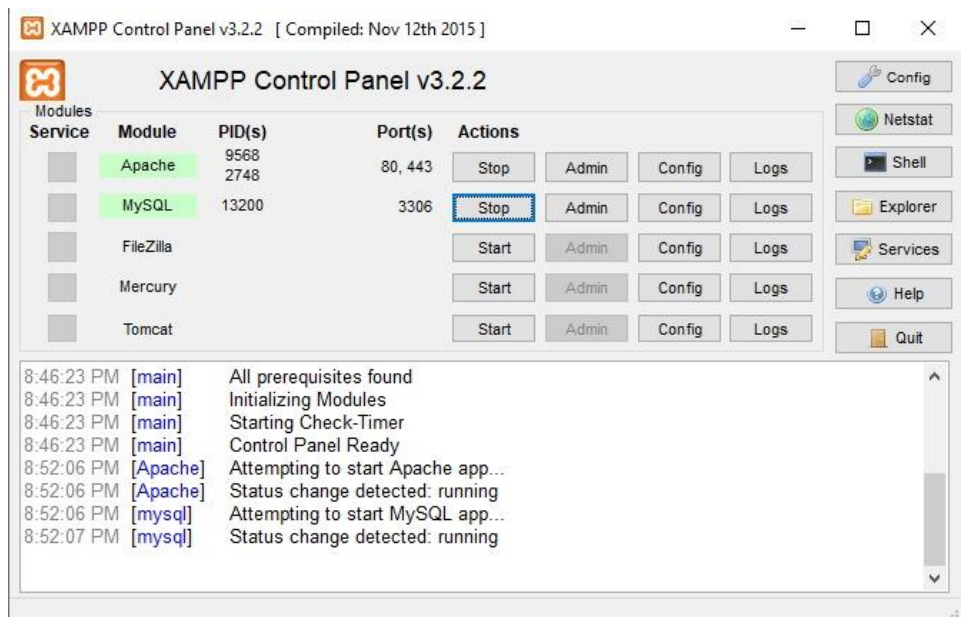
3. kemudian *extract* file yang telah berhasil di *download*.



4. Setelah itu pindahkan folder yang telah di *extract* tadi kedalam folder htdocs (C:\xampp\htdocs)

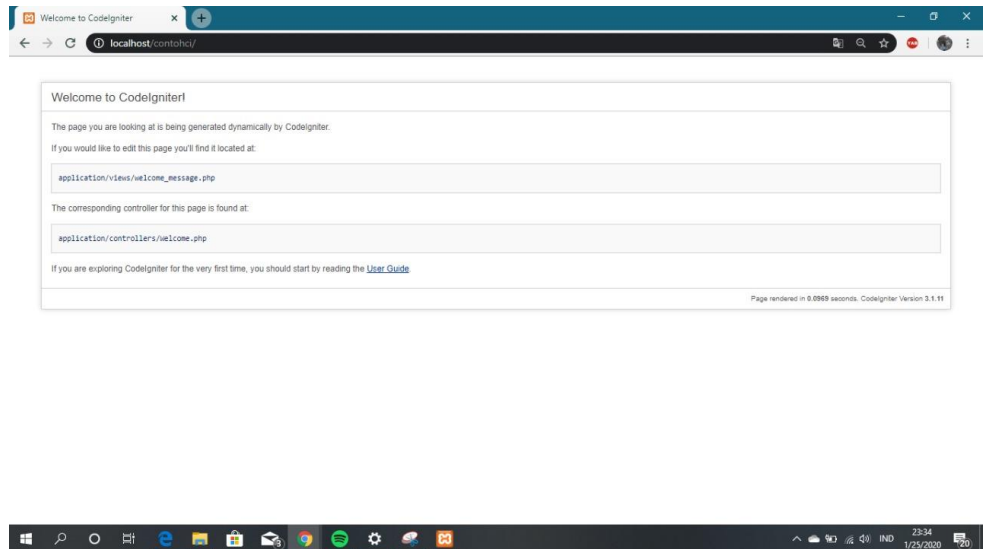


5. Kemudian ubah nama file tersebut sesuai dengan yang anda inginkan, disini saya menggunakan nama file contohCI.
6. Setelah itu buka aplikasi Xampp dan jalankan xampp dengan mengklik pada bagian *actions* sampai berubah menjadu *stop*.



7. Selanjutnya buka *browser* dan ketikkan

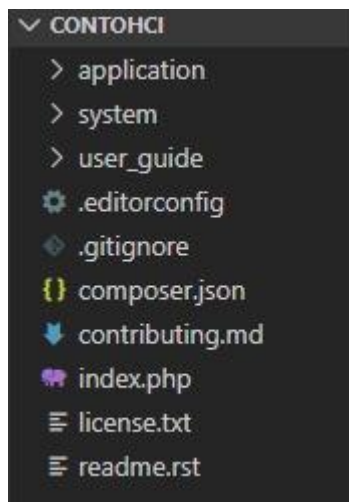
`http://localhost/contohci/`



8. Selamat *codeigniter* berhasil di install.

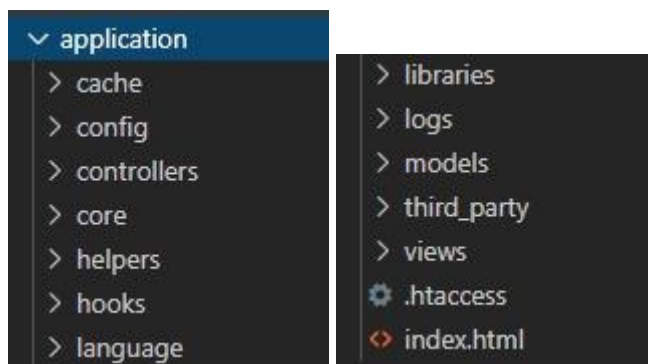
9. Setelah itu buka VisualCode yang telah di install tadi, hal ini dilakukan untuk mengenali struktur direktori pada *codeigniter*.

Berikut adalah struktur direktori *codeigniter*



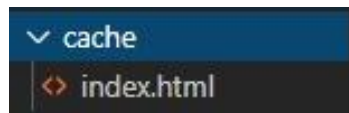
Terdapat dua direktori penting didalam CI yaitu folder *application* dan folder *System* . selain juga terdapat direktori folder *user_guide* dan beberapa file. Berikut adalah beberapa penjelasan yang terdapat dalam CI:

- Folder *Application*

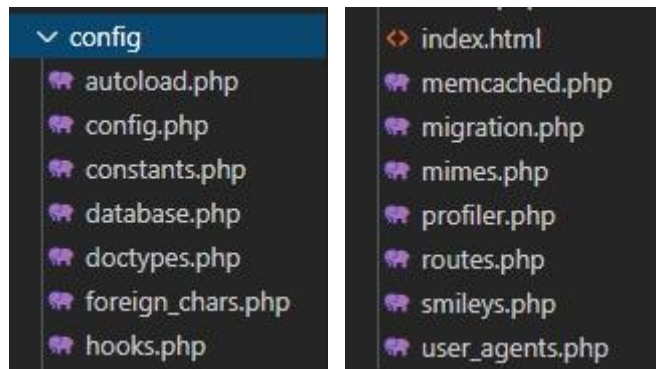


Folder *application* berisi code aplikasi. Disinilah kita akan menulis semua baris kode aplikasi yang akan kita bangun. Dalam folder *application* terdapat beberapa direktori yaitu:

- Folder *chace*, adalah suatu folder yang berisi *chace* dari aplikasi.

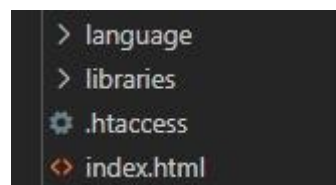
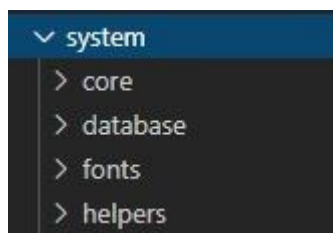


- Folder *config*, adalah suatu folder yang berisi konfigurasi aplikasi.

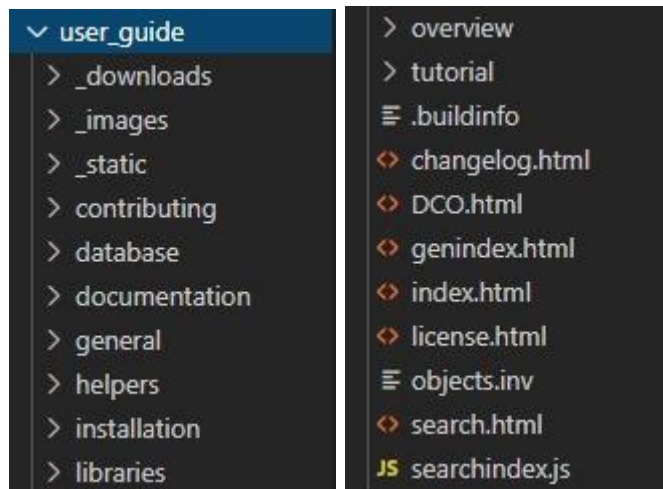


- Autoload.php*, merupakan suatu tempat yang mendefinisikan *autoload*.
- Config.php*, adalah suatu file untuk mengkonfigurasi aplikasi.
- Constants.php*, merupakan suatu file yang berisi konstanta.
- Database.php*, merupakan suatu file yang berisi konfigurasi database aplikasi.
- Doctypes.php*, adalah suatu file yang berisi definisi untuk *doctype html*.
- Foreign_chars.php*, merupakan suatu file yang berisi *character* dan *symbol*.
- Hooks.php*, merupakan suatu file yang berisi konfigurasi *hooks*.

- h. *Index.html*, merupakan suatu file untuk mencegah *direct access*.
 - i. *Memchaced.php* , merupakan suatu file yang berisi suatu konfigurasi untuk *menchace*.
 - j. *Migration.php*, merupakan suatu file yang berisi untuk migrasi.
 - k. *Mimes.php*, merupakan suatu file yang berisi definisi type file
 - l. *Profiler.php*, merupakan suatu file yang berisi konfigurasi untuk *profiler*.
 - m. *Routes.php*, merupakan suatu file yang digunakan sebagai tempat untuk menulis *route* aplikasi.
 - n. *Smileys.php*, berisi kode untuk emoji.
 - o. *User_agents.php*, merupakan suatu file yang berisi definisi untuk *user_agents*.
- Folder *System*, Folder ini berisi kode-kode inti dari *codeigniter*. Perlu diingat bahwa janganlah anda mengubah apapun didalam direktori ini.jika anda ingin *menupgrade* versi ke yang baru anda cukup *mereplace* direktori tersebut dengan yang baru.



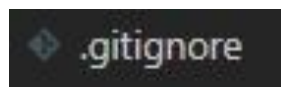
- Folder *User-guide*, Folder ini berisi dokumentasi lengkap dari *codeigniter*, dengan folder ini kita bisa mengakses dokumentasi CI secara *offline*. Anda bisa menghapus direktori ini saat *web* yang anda buat telah jadi.



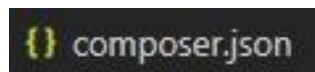
- File *.editorconfig*, File ini berisi konfigurasi untuk *text* editor.



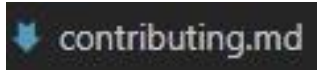
- File *.gitignore*, File ini berisi daftar file dan folder yang akan diabaikan oleh git.



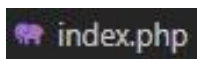
- File *Composer .json*, File ini adalah suatu yang berisi keterangan project dan keterangan *library* yang digunakan, selain itu file ini dibutuhkan oleh composer



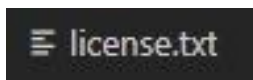
- File *Contributing.md*, file ini adalah suatu file yang berisi penjelasan tentang cara berkontribusi diproyek CI. File ini dapat dihapus apabila *web* sudah jadi.



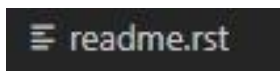
- File *Index.php*, file ini adalah suatu file utama dari *codeigniter* yang merupakan file yang akan dibuka pertama kali saat kita mengakses *web*.



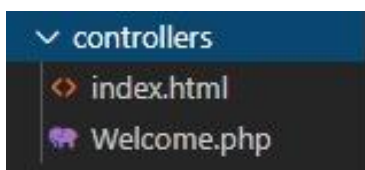
- File *license.txt*, file ini berisi keterangan lisensi dari *codeigniter*



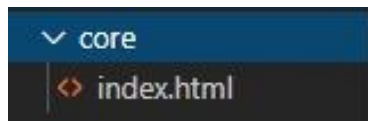
- File *Readme.rst*, File ini sama seperti dengan file *contributing.md*. file ini berisi penjelasan dan informasi tentang proyek *codeigniter*. File ini dapat dihapus saat *web* sudah selesai.



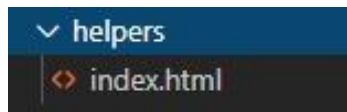
- Folder *Controller*, merupakan suatu folder yang berisi kode-kode *controller*.



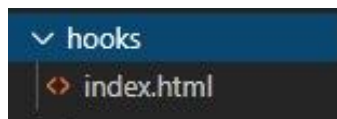
- Folder *Core*, merupakan suatu folder yang berisi kode-kode untuk *coustome core*.



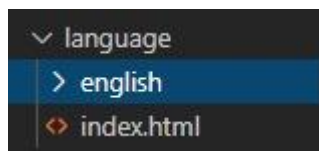
- Folder *Helpers*, merupakan suatu folder adalah fungsi-fungsi helper.



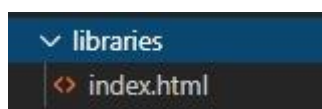
- Folder *Hooks*, merupakan suatu folder yang berisi kode untuk *script hook*.



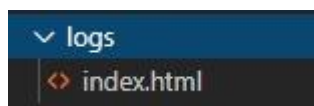
- Folder *language*, merupakan suatu folder yang berisi *string* untuk bahasa apabila web mendukung multi bahasa.



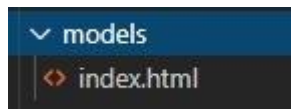
- Folder *Libraries*, merupakan suatu folder yang berisi *library*.



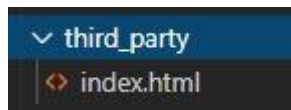
- Folder *Logs*, merupakan suatu folder yang berisi *logs* dari aplikasi.



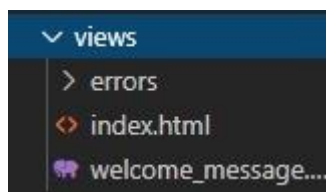
- Folder *models*, berisi kode-kode untuk model



- Folder *Third_party*, merupakan suatu folder yang berisi *library* dari pihak ketiga

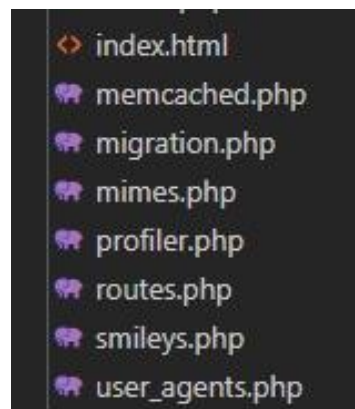
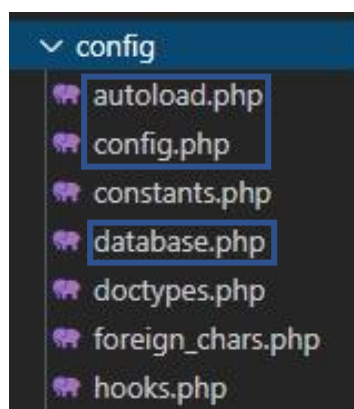


- Folder *Views*, merupakan suatu folder yang berisi kode suatu *view*.



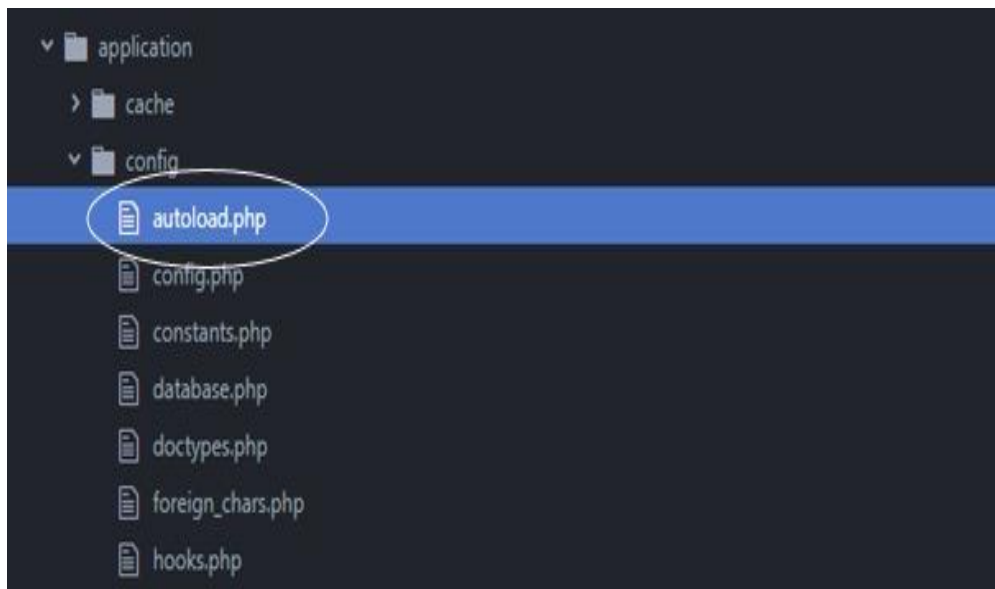
4.1.6 Konfigurasi Dasar *CodeIgniter*

Dalam memulai *codeigniter* ada beberapa konfigurasi yang harus di lakukan pada beberapa file yaitu file *autoload.phph*, file *config*, dan *database.php*. Semua konfigurasi tersebut terletak didalam folder *application/config*.



- *Autoload.php*

Pada file ini digunakan untuk mengatur fungsi-fungsi yang akan di muat secara otomatis diawal ketika suatu program dijalankan. Untuk melakukan konfigurasi pada file *autoload.php* silahkan buka folder *application/config/autoload.php*.



Ada beberapa hal yang bisa di *load* secara otomatis dalam *codeigniter* yaitu *package*, *libraries*, *drivers*, *helper files*, *custom config files*, *language files*, dan *models*.

Untuk konfigurasi dasar yang perlu anda ketahui adalah *libraries* dan *helper files* hal tersebut memiliki tujuan agar beberapa *library* dan *helper* tertentu berjalan otomatis.

Untuk melakukan konfigurasi pada *libraries* buka file *autoload.php* dengan *text editor* kemudia temukan kode berikut :

```
$autoload['libraries'] = array();
```

Setelah itu atur menjadi seperti gambar berikut:

```
$autoload['libraries'] = array('database');|
```

Pada kode diatas artinya kita akan *meload library database* secara otomatis, sehingga anda dapat menggunakan fungsi-fungsi database pada *codeigniter*.

Selanjutnya kita akan melakukan konfigurasi pada *helper files*, kemudian temukan kode berikut:

```
$autoload['helper'] = array();
```

Setelah itu atur menjadi seperti gambar berikut:

```
$autoload['helper'] = array('url');
```

Pada kode diatas artinya kita akan *meload helper URL* secara otomatis. Dengan demikian anda dapat menggunakan fungsi-fungsi URL pada *codeigniter*, misalnya fungsi *base_url()*, *site_url()*, *URI Segment*, dan sebagainya.

- *Config.php*

Pada file ini terdapat beberapa konfigurasi yang secara standar sudah terkonfigurasi, namun terdapat beberapa konfigurasi yang perlu diperhatikan:

- *Base_url*, merupakan url dasar dari *project* anda. Temukan kode berikut:

```
$config['base_url'] = '';
```

Kemudian atur menjadi gambar berikut:

```
$config['base_url'] = 'http://localhost/contohci/';
```

- *Database.php*

Dilihat dari nama filenya anda pasti sudah mengetahui apa fungsi dari file ini. File *database.php* berfungsi untuk melakukan konfigurasi yang berkaitan dengan konfigurasi database dari *website* yang akan di buat. Konfigurasi yang perlu diperhatikan tersebut diantaranya: *hostname*, *username*, *password*, dan *database*.

Untuk melakukan konfigurasi pada *database.php* buka file *database.php* dengan *text* editor.



Kemudian temukan kode berikut :

```

$active_group = 'default';
$query_builder = TRUE;

$db['default'] = array(
    'dsn' => '',
    'hostname' => 'localhost',
    'username' => '',
    'password' => '',
    'database' => '',
    'dbdriver' => 'mysqli',
    'dbprefix' => '',
    'pconnect' => FALSE,
    'db_debug' => (ENVIRONMENT !== 'production'),
    'cache_on' => FALSE,
    'cachedir' => '',
    'char_set' => 'utf8',
    'dbcollat' => 'utf8_general_ci',
    'swap_pre' => '',
    'encrypt' => FALSE,
    'compress' => FALSE,
    'stricton' => FALSE,
    'failover' => array(),
    'save_queries' => TRUE
);

```

Setelah itu ubah kode tersebut seperti gambar dibawah ini :

```

$active_group = 'default';
$query_builder = TRUE;

$db['default'] = array(
    'dsn' => '',
    'hostname' => 'localhost',
    'username' => 'root',
    'password' => '',
    'database' => 'contohci',
    'dbdriver' => 'mysqli',
    'dbprefix' => '',
    'pconnect' => FALSE,
    'db_debug' => (ENVIRONMENT !== 'production'),
    'cache_on' => FALSE,
    'cachedir' => '',
    'char_set' => 'utf8',
    'dbcollat' => 'utf8_general_ci',
    'swap_pre' => '',
    'encrypt' => FALSE,
    'compress' => FALSE,
    'stricton' => FALSE,
    'failover' => array(),
    'save_queries' => TRUE
);

```

4.2 Bahasa Pemrograman Yang Dapat Digunakan

4.2.1 PHP (*Hypertext Preprocessor*)

PHP adalah singkatan dari *Hypertext Preprocessor* yang merupakan bahasa pemrograman yang bersifat *open source* yang berada didalam server yang diproses di server. PHP salah satu bahasa pemrograman yang memiliki script yang terintegrasi dengan HTML dan berada pada server (*server side HTML embedded scripting*).[9]

PHP merupakan bahasa pemrograman pelengkap HTML yang memungkinkan aplikasi web dinamis untuk pengolahan data, pemrosesan data dari *user via form*, membuat buku tamu, toko online, dan lain sebagainya. Dengan mudah PHP dapat melakukan koneksi ke database karen PHP memang dilengkapi fitur yang memungkinkan koneksi ke PHP dilakukan dengan mudah tanpa harus memusingkan (Tim EMS, 2016:55).

PHP adalah bahasa *server-side scripting* yang menyatu dengan HTML untuk membuat halaman web yang dinamis. Maksud dari *server-side scripting* adalah sintaks dan perintah-perintah yang diberikan akan sepenuhnya akan dijalankan diserver tetapi disertakan pada dokumen HTML. Pembuatan web ini merupakan kombinasi antara php sendiri sebagai bahasa pemrograman dan HTML sebagai pembangun halaman web (Bimo sunarfrihantono, ST 2002:9).



Gambar Hypertext Preprocessor

Sumber : <https://www.webhozz.com/>

Menurut Andi Pramono dan M. Syafii (2004 : 2), PHP atau kependekan dari *Hypertext Preprocessor* adalah sebuah bahasa pemrograman berbasis *web* yang mempunyai banyak keunggulan dibandingkan dengan bahasa pemrograman berbasis *web* yang lain. PHP merupakan bahasa pemrograman yang bersumber dari Perl. Sedangkan Perl merupakan pengembangan dari bahasa C.

Oleh karenanya, struktur pemrograman yang ada di PHP sama dengan yang ada di bahasa C. Melihat bahwa PHP merupakan pengembangan dari bahasa C secara tidak langsung, maka PHP mempunyai banyak sekali fitur-fitur yang dapat digunakan. Misalnya, PHP dapat mengakses shell di Linux, mempunyai fungsi yang lengkap berhubungan dengan *networking*.

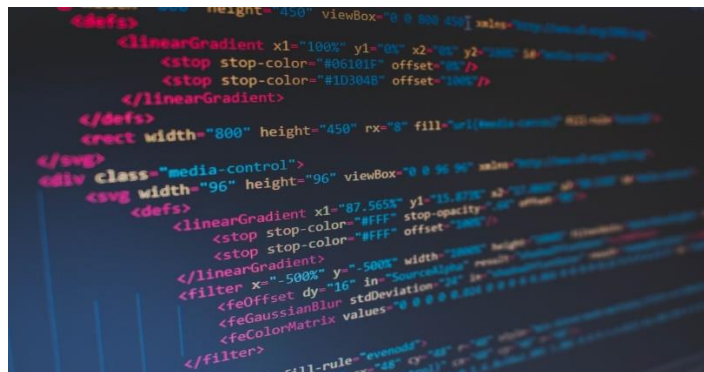
Salah satu keunggulan PHP dibanding bahasa pemrograman lainnya adalah PHP dapat diperoleh secara gratis, meskipun bukan berarti karena gratis kemampuannya menjadi pas-pasan. PHP sangat powerful. Terbukti dengan banyaknya website yang dibangun menggunakan PHP.

PHP juga terkenal lebih aman daripada bahasa pemrograman website yang lain. PHP juga sudah mendukung OOP (*Object Oriented Programming*) sehingga *maintenance* kode menjadi jauh lebih mudah dibandingkan procedural.

PHP bisa berinteraksi dengan database, file dan folder, PHP ini termaksud bahasa *cross-platform*, yang mana PHP ini bisa berjalan di sistem operasi yang berbeda-beda (Windows, Linux, ataupun MAC).[10]

Berdasarkan pernyataan diatas dapat disimpulkan bahwa PHP adalah bahasa adalah sebuah bahasa pemrograman yang sangat fleksibel untuk digunakan dalam membuat sebuah website, dan juga PHP dapat dijalankan di bawah sistem operasi LINUX dan Windows.

4.2.2 HTML



Gambar Script HTML

Sumber : enimamedia.com

HTTP merupakan protokol yang digunakan untuk mentransfer data antara *web server* ke *web browser*. Protokol ini mentransfer dokumen-dokumen web yang ditulis atau berformat HTML. Dikatakan markup karena HTML berfungsi untuk "mepercantik" file text biasa untuk ditampilkan pada program *web browser*. Hal ini dilakukan dengan menambahkan tag-tag (perintah khusus) pada file teks biasa tersebut.

Tag HTML biasanya berupa tag-tag yang berpasangan dan ditandai dengan simbol `<dan>`. Pasangan dari sebuah tag ditandai dengan tanda `'/'`. misalnya awalnya perintah tagnya `<contoh>` maka diakhiri dengan `</contoh>`.

Supaya dapat menghasilkan tampilan wujud yang terintegrasi Pemformatan hiperteks sederhana ditulis dalam berkas format ASCII sehingga menjadi halaman web dengan perintah-perintah HTML.

HTML merupakan sebuah bahasa yang bermula bahasa yang sebelumnya banyak dipakai di dunia percetakan dan penerbitan yang disebut Standard Generalized Markup Language (SGML).

Sekarang ini HTML merupakan standar Internet yang dikendalikan dan didefinisikan pemakaiannya oleh World Wide Web Consortium (W3C). Pada tahun 1989, HTML dibuat oleh kolaborasi Berners-lee Robert dengan Caillau TIM pada saat mereka bekerja di CERN (CERN merupakan lembaga penelitian fisika energi tinggi di Jenewa).

4.2.2.1 Fungsi HTML (*HyperText Markup Language*)

HTML (*HyperText Markup Language*) adalah suatu bahasa yang menggunakan tanda-tanda tertentu (*tag*) untuk menyatakan kode-kode yang harus ditafsirkan oleh *browser* agar halaman tersebut dapat ditampilkan secara benar.

Secara umum, fungsi HTML adalah untuk mengelola serangkaian data dan informasi sehingga suatu dokumen dapat diakses dan ditampilkan di Internet melalui layanan *web*.

Fungsi HTML yang lebih spesifik yaitu :

- Membuat halaman *web*.
- Menampilkan berbagai informasi di dalam sebuah *browser Internet*.
- Membuat link menuju halaman web lain dengan kode tertentu (*hypertext*).
- HTTP atau *Hypertext Transfer* Protokol merupakan protokol yang digunakan untuk mentransfer data atau document yang berformat HTML dari *web server* ke *web browser*. Dengan HTTP inilah yang memungkinkan Anda menjelajah internet dan melihat halaman *web*.

4.2.2.2 Sejarah HTML

HTML dibuat oleh Tim Berners-Lee, seorang ahli fisika di lembaga penelitian CERN yang berlokasi di Swiss. Dia memiliki ide tentang sistem *hypertext* yang berbasis internet.

Hypertext merujuk pada teks yang memuat referensi (link) ke teks lain yang bisa diakses langsung oleh viewer. Tim merilis versi pertama HTML pada tahun 1991, dan di dalamnya terdiri atas 18 HTML tag. Sejak saat itu, setiap kali bahasa HTML

merilis versi teranyarnya, selalu ada *tag* dan *attribute* (*tag modifier*) terbaru.

Berdasarkan *HTML Element Reference* milik Mozilla *Developer Network*, untuk saat ini, ada 140 HTML tag meskipun sebagiannya sudah usang (tidak lagi didukung oleh versi terbaru *browser*).

Berkat popularitasnya yang terus meningkat, HTML kini dianggap sebagai web standard yang resmi. Spesifikasi HTML di-*maintain* dan dikembangkan oleh *World Wide Web Consortium* (W3C). Cek versi terbaru dari bahasa ini di website W3C.

Upgrade HTML besar-besaran terjadi pada tahun 2014, dan hasilnya adalah pengenalan HTML5. Pada *upgrade* tersebut, terdapat semantic baru yang memberitahukan arti dari kontennya sendiri, seperti `<article>`, `<header>`, dan `<footer>`.

4.2.2.3 Cara Kerja HTML

Dokumen HTML adalah file yang diakhiri dengan ekstensi `.html` atau `.htm`. Ekstensi file ini bisa dilihat dengan menggunakan *web browser* apa pun (seperti Google Chrome, Safari, atau Mozilla Firefox). *Browser* tersebut membaca file HTML dan *me-render* kontennya sehingga user internet bisa melihat dan membacanya.

Biasanya, rata-rata situs web menyertakan sejumlah halaman HTML yang berbeda-beda. Contohnya, beranda utama, halaman ‘tentang kami’, halaman kontak yang semuanya memiliki dokumen HTML terpisah.

Masing-masing halaman HTML terdiri atas seperangkat tags (bisa disebut juga *elements*), yang mengacu pada *building block*

halaman website. Tag tersebut membuat hirarki yang menyusun konten hingga menjadi bagian, paragraf, heading, dan *block* konten lainnya.

Sebagian besar element HTML memiliki tag pembuka dan penutup yang menggunakan syntax `<tag></tag>`.

Berikut contoh kode dari susunan atau struktur HTML:

```
<div>
  <h1>The Main Heading</h1>
  <h2>A catchy subheading</h2>
  <p>Paragraph one</p>
  
  <p>Paragraph two with a <a
href="https://example.com">hyperlink</a></p>
</div>
```

Elemen teratas dan terbawah adalah division sederhana (`<div></div>`) yang bisa Anda gunakan untuk mark up bagian konten yang lebih besar.

Susunan HTML di atas terdiri atas heading (`<h1></h1>`), subheading (`<h2></h2>`), dua paragraf (`<p></p>`), dan satu gambar (``).

Paragraf kedua meliputi sebuah link (`<a>`) dengan attribute href yang terdiri atas URL tujuan.

Tag gambar memiliki dua attribute, src untuk path gambar dan alt untuk deskripsi gambar.

4.2.2.4 Gambaran Umum Tentang Tag HTML yang Paling Sering Digunakan

Tag HTML memiliki dua tipe utama: *block-level* dan *inline tags*. Elemen *block-level* memakai semua space yang tersedia dan selalu membuat line baru di dalam dokumen. Contoh dari *tag block* adalah *heading* dan paragraf.

Elemen *inline* hanya memakai *space* sesuai dengan kebutuhannya dan tidak membuat line baru di halaman. Biasanya elemen ini akan memformat isi konten dari elemen *block-level*. Contoh dari *tag inline* adalah link dan *emphasized strings*.

- *Tag Block-Level*

Tiga *tag block-level* yang harus dimiliki oleh setiap dokumen HTML adalah `<html>`, `<head>`, dan `<body>`.

Tag `<html></html>` adalah elemen level tertinggi yang menyertakan setiap halaman HTML.

Tag `<head></head>` menyimpan informasi meta, seperti judul dan charset halaman.

Tag `<body></body>` melampirkan semua konten yang muncul pada suatu halaman.

```
<html>
  <head>
    <!-- META INFORMATION -->
  </head>
  <body>
    <!-- PAGE CONTENT -->
  </body>
</html>
```

Heading memiliki 6 level di HTML. Level tersebut bervariasi, mulai dari `<h1></h1>` sampai ke `<h6></h6>`, di mana h1 merupakan level heading tertinggi dan h6 adalah level terendah. Paragraf dibuka dan ditutup dengan tag `<p></p>`, sedangkan *blockquote* menggunakan tag `<blockquote></blockquote>`.

Division merupakan bagian konten yang lebih besar dan biasanya terdiri atas beberapa paragraf, gambar, kadang-kadang *blockquote*, dan elemen lebih kecil lainnya. Kita bisa membuat markup dengan menggunakan tag `<div></div>`. Di dalam elemen div juga terdapat tag div lainnya.

Anda juga bisa menggunakan tag `` untuk list yang berurutan dan `` untuk list yang tidak berurutan. Masing-masing list item harus dibuka dan ditutup dengan tag ``. Sebagai contoh, di bawah ini adalah tampilan dasar dari list tidak berurutan dalam HTML:

```
<ul>
  <li>List item 1</li>
  <li>List item 2</li>
  <li>List item 3</li>
</ul>
```

- *Tag Inline*

Sebagian besar *tag inline* digunakan untuk memformat teks. Sebagai contoh, tag `` akan *render* elemen ke format *bold*, sedangkan tag `` akan ditampilkan dalam format *italic*.

Hyperlink adalah elemen *inline* yang mewajibkan adanya tag `<a>` dan attribute href untuk mengindikasikan tujuan link:

```
<a href="https://example.com/">Click me!</a>
```

Gambar (*image*) juga merupakan elemen *inline*. Anda dapat menambahkan satu gambar dengan menggunakan `` tanpa harus membubuhkan *tag* penutup. Hanya saja, Anda disarankan menggunakan *attribute* `src` untuk menentukan path gambar, misalnya:

```

```

Jika ingin mempelajari lebih dalam tentang tag HTML, silakan baca artikel kami tentang *HTML cheat sheet* (kami juga menyediakan link untuk diunduh).

4.2.2.5 Kelebihan dan Kekurangan HTML

Sama seperti hal teknis lainnya dalam dunia web, HTML juga punya kelebihan dan kekurangannya.

- Kelebihan
 - Bahasa yang digunakan secara luas dan memiliki banyak sumber serta komunitas yang besar.
 - Dijalankan secara alami di setiap *web browser*.
 - Memiliki *learning curve* yang mudah.
 - *Open-source* dan sepenuhnya gratis.
 - Bahasa markup yang rapi dan konsisten.
 - *Standard web* yang resmi di-maintain oleh *World Wide Web Consortium* (W3C).
 - Mudah diintegrasikan dengan bahasa *backend*, seperti PHP dan Node.js.

- Kekurangan

Paling sering digunakan untuk halaman *web* statis. Untuk fitur dinamis, Anda bisa menggunakan *JavaScript* atau bahasa backend, seperti PHP.

HTML tidak memungkinkan user untuk menjalankan logic. Alhasil, semua halaman web harus dibuat terpisah meskipun menggunakan elemen yang sama, seperti header dan footer.

Fitur-fitur baru tidak bisa digunakan secara cepat di sebagian browser.

Terkadang perilaku browser susah untuk diprediksi (misalnya, browser lama tidak selalu bisa render tag yang lebih baru).

4.2.3 CSS (*Cascading Style Sheets*)

CSS (*Cascading Style Sheets*) Pada versi HTML yang terdahulu, web browser mengontrol tampilan (rendering) dari setiap halaman web. Jika menggunakan elemen H1 pada (*large heading*) pada web dokumen, browser akan merender elemen tersebut.

Dengan adanya CSS, *programmer* dapat mengontrol bagaimana *browser me-render* halaman web. Mengaplikasikan CSS pada halaman web dapat memberikan tampilan yang lebih menarik dan spesifik sesuai dengan tema pada sebuah web site.

Teknologi CSS memberikan fasilitas untuk menentukan style (misal; *spacing, margins*) dari elemen halaman web terpisah dari struktur dokumen web (*section headers, body text, links*). Pemisahan tersebut memberikan peningkatan yang lebih besar dalam pengaturan *web pages*, dan membuat perubahan –

perubahan *style* dalam dokumen dapat dilakukan lebih cepat dan lebih mudah.

4.2.3.1 Kelebihan dan Kekurangan CSS

a. Kelebihan

- *Update* tampilan lebih mudah
- Beban *bandwidth* lebih kecil
- Modifikasi *Web template* lebih mudah
- Lebih mudah digunakan pada *mobile phone*
- *Search engine friendly*
- Memisahkan desain dengan konten halaman web
- Mengatur desain seefisien mungkin
- Jika kita ingin mengubah suatu tema halaman web, cukup modifikasi pada css saja
- Menghadirkan sesuatu yang tidak dapat dilakukan oleh HTML
- Lebih mudah didownload karena lebih ringan ukuran filenya
- Satu CSS dapat digunakan banyak halaman web
- Menghemat penulisan kode, karena satu css dapat dipakai beberapa kali
- Mempersingkat waktu kerja, baik saat membuat maupun saat modifikasi halaman *Web/Blog*, dan masih banyak lagi.

b. Kekurangan

- Tampilan pada browser berbeda-beda.
- Kadang juga terdapat *browser* yang tidak support CSS (*browser* lama).
- Harus tahu cara menggunakannya.

- Dibutuhkan waktu lebih lama dalam membuatnya.
- Belum lagi ada *bug/error* dalam CSS

BAB V

BASIS DATA

5.1 Konsep Basis Data

Jika misalnya kita memiliki banyak sekali buku dan anggap saja kita punya dua solusi untuk menyimpannya. Solusi pertama adalah menyimpan di sembarang tempat dan yang kedua disimpan di dalam rak yang tersusun rapi dengan kode rak dan seterusnya. Pada saat tersimpan solusi pertama tampak sangat memudahkan terserah kita mau simpan dimana namun pada saat penjualan buku, akan sangat sulit bagi kita untuk menemukan buku yang kita cari karena kita menyimpannya tidak beraturan. berbeda dengan solusi kedua, Pada saat penyimpanan masih diperlukan waktu yang sedikit lebih lama tapi kelebihan waktunya tidak signifikan jika dibandingkan dengan kemudahan pada saat pencarian. Dengan keteraturan pada saat penyimpanan mengakibatkan kemudahan dan kecepatan pada saat pencarian kembali buku yang disimpan

Nah, itulah kira-kira analogo betapa bermanfaatnya basis data. Buku itu kita ibaratkan sebagai data baris rak itu, kita ibaratkan sebagai tabel dan rak buku itu kita ibaratkan sebagai basis data.

5.2 Definisi Basis Data

Basis data dapat didefinisikan sebagai himpunan kelompok data yang saling berhubungan yang diorganisasikan sedemikian rupa agar kelak dapat dimanfaatkan kembali dengan cepat dan mudah. prinsip utamanya adalah pengaturan data. tujuan utamanya kemudahan dan kecepatan dalam pengambilan kembali data.

5.3 Tujuan Basis Data

Secara lebih lengkap pemanfaatan basis data dilakukan untuk memenuhi tujuan berikut ini

- Kecepatan dan kemudahan (*speed*)
- Profisiensi ruang penyimpanan (*space*)
- Keakuratan (*accuracy*)
- Ketersediaan (*availability*)
- Kelengkapan (*completeness*)
- Keamanan (*security*)
- Pemakaian bersama (*sharability*)

5.4 Penggunaan Basis Data

Setiap organisasi atau perusahaan maka untuk mencapai efisiensi, daya saing, dan kecepatan operasional organisasi atau perusahaan digunakanlah aplikasi sistem informasi yang biasanya menggunakan perangkat komputer. Basis data adalah suatu komponen utama dalam sistem informasi dan tidak ada sistem informasi yang bisa dijalankan tanpa adanya basis data. jadi untuk perusahaan menggunakan aplikasi berbasis komputer maka basis data mutlak diperlukan. konkritnya, beberapa pemanfaatan basis data antara lain

- Kepegawaian untuk berbagai perusahaan yang memiliki banyak pegawai.
- Pergudangan atau *inventory* dan penjualan untuk perusahaan manufaktur atau pabrik, supermarket, apotek dan lain-lain.

- Akuntansi, untuk Bank dan perusahaan-perusahaan yang melibatkan uang.
- Reservasi, untuk hotel, pesawat, kereta api
- Catatan medik untuk rumah sakit
- Akademik, untuk perguruan tinggi atau sekolah.
- Penggajian, untuk perusahaan yang memiliki pegawai yang cukup banyak dll.

5.5 Perancangan Basis Data

Sebelum kita membuat basis data terlebih dahulu dilakukan perancangan. proses perancangan ini bersifat konseptual. Kita belum mengetahui DBMS apa yang akan kita gunakan untuk mengimplementasikan rancangan basis data yang dibuat. Tujuan perancangan basis data adalah mendapatkan skema basis data yang meminimalisasi terjadinya redundansi dan duplikasi data serta menjaga integrasi data. Kebanyakan metode perancangan berbasis pada model basis data relasional. Pada basis data relasional, dan diatur melalui pembuatan tabel-tabel dan terdapat keterkaitan antar tabel yang satu dengan lainnya (relasi).

Salah satu pemodelan yang sering digunakan untuk merancang basis data relasional adalah *Entity Relationship Diagram*. Dua elemen fundamental pada ER adalah entitas dan *relationship* (keterhubungan).

6.5.1 Entitas

Entitas adalah suatu objek baik itu nyata maupun abstrak di dunia nyata yang dapat dibedakan dari objek lain berdasarkan karakteristik yang dimilikinya. Misalnya : pegawai perusahaan, barang yang dijual ditoko, mahasiswa, dosen, mata kuliah, jurusan, layanan suatu bank, dan sebagainya. Dalam basis data, kita harus menentukan entitas apa saja yang diinformasinya perlu disimpan. Pada ER Diagram, entitas memiliki symbol persegi Panjang.

Informasi yang ingin disimpan dari suatu entitas disebut dengan *property* . misalnya pada entitas pada mahasiswa, beberapa *property* yang mungkin adalah NPM, nama mahasiswa, tanggal lahir, alamat, nomor telepon, dan lainnya. Pada entitas barang, beberapa atribut yang mungkin adalah nomor identitas barang, nama barang, jumlah barang, harga beli, dan harga jual. Pada ER Diagram atribut dilambangkan dengan elips.

Untuk membedakan data pada suatu entitas, biasanya adalah suatu atribut digunakan sebagai *identifier* yang dapat membedakan antara satu data dengan lainnya. Atribut ini disebut dengan *primary key*. *Primary key*, karena NPM merupakan identitas unik dari seorang mahasiswa (tidak ada mahasiswa yang memiliki NPM sama). Pada kasus entitas barang, biasanya setiap barang memiliki nomor identitas barang yang digunakan sebagai *primary key*. Pada entitas mata

kuliah, biasanya digunakan kode mata kuliah sebagai *primary key*.

Pertimbangan pemilihan atribut mana yang dijadikan sebagai *primary key* tergantung dari beberapa hal, antara lain:

- Atribut harus unik. Itu artinya untuk objek/barang yang berbeda atribut tersebut pasti beda juga.
- Nilai atribut tersebut jarang berubah.
- Mudah diingat dan biasanya memiliki format konsisten.

NIM	Nama_mhs	JK	Alamat	Tgl_Lahir	No_tlp
117318274	Nurul Izza	P	Jln. Sariasih	18/10/1999	0891638119
			II		
114890490	Aulyardha	P	Jln. Sariasih	23/09/1999	08531689190
			II		
.....

Untuk tiap atribut, nilai atribut memiliki batasan yang disebut dengan domain constraint. misalnya, atribut merupakan string dengan panjang maksimal 9. jenis kelamin hanya boleh diisi dengan L atau P. atribut mana yang harus diisi dan mana yang tidak boleh diisi. jika suatu atribut nilainya tidak atau belum terdefinisi Dan diperbolehkan untuk tidak diisi dahulu, maka atribut tersebut adalah diberi nilai null. misalnya seorang mahasiswa tidak memiliki telepon,

maka nilai kolom masih tersebut adalah *null*. contoh lainnya, misalkan alamat seorang mahasiswa belum diketahui maka kita bisa memberikan nilai null.

Pada beberapa kasus, suatu atribut dapat mengacu atau merefer ke atribut pada tabel lain (atau atribut lain pada tabel itu sendiri). atribut yang diracun oleh atribut lain disebut *foreign key*. *Foreign key* adalah suatu cara untuk menjaga integritas data.

6.5.2 Relasi

Relasi adalah sosialisasi yang menyatakan keterhubungan antar entitas. misalnya:

- Relasi mengajar yang menghubungkan antara entitas dosen dengan mata kuliah. Relasi ini menyatakan bahwa seorang dosen hanya boleh mengajar banyak mata kuliah dan satu mata kuliah bisa diajar oleh beberapa orang dosen.
- Relasi tempat antara entitas mata kuliah dengan ruangan. relasi ini menyatakan bahwa satu mata kuliah bisa diajarkan di beberapa ruang yang berbeda- beda dan satu ruangan bisa digunakan untuk mengajar mata kuliah yang berbeda-beda
- Relasi nilai, yang menghubungkan antara entitas mahasiswa dengan mata kuliah. Relasi ini menyatakan bahwa 1 orang mahasiswa memiliki satu nilai untuk satu

mata kuliah, namun satu mata kuliah bisa memiliki banyak nilai mahasiswa.

6.5.3 Kardinalitas

Relasi memiliki derajat keterhubungan. hubungan antar hubungan antar entitas pada suatu relasi seperti pada contoh diatas disebut dengan kardinalitas. Terdapat tiga jenis kardinalitas

- Satu ke satu, misalnya relasi registrasi
- Satu ke banyak, atau banyak ke-1. misalnya relasi tempat
- Banyak-banyak, misalnya relasi nilai.

contoh menentukan kardinalitas :

Misalnya saja di suatu kampus yang boleh menjadi admin adalah dosen

- Yang jadi pertanyaan, berapa maksimal dosen yang boleh menggunakan satu *login* admin tertentu? jawabannya 1. Bolehkah sebuah *login* admin tidak dimiliki oleh dosen (tidak Bertuan?) jawabannya tidak boleh karena munculnya sebuah admin adalah karena ada dosen yang registrasi sebagai admin (minimal 1)
- Berapa maksimal *login* admin yang boleh dimiliki oleh seorang dosen? jawabannya 1, Bolehkah dosen tidak menjadi admin? jawabannya boleh

Kardinalitas sebuah relasi diambil dari nilai maksimalnya sehingga nilai relasi registrasi mempunyai kardinalitas satu kurang satu.

6.5.4 Langkah-langkah Membuat ER Diagram

Langkah-langkah membuat er diagram dengan studi kasus *website* akademik. Identifikasi dan tetapkan seluruh himpunan entitas yang akan terlibat. misalnya studi kasus yang kita ambil adalah *website* akademik. objek-objek yang terlibat dengan proses akademik antara lain : dosen, mahasiswa, mata kuliah, ruangan, jadwal, dan admin.

Tentukan atribut-atribut key dari masing-masing himpunan entitas. atribut key adalah atribut yang menjadi Identifier (pembeda atau identitas) dalam suatu entitas. dosen Keynya adalah NIP, mahasiswa keynya adalah NIM.

Identifikasi dan tetapkan seluruh relasi di antara entitas yang ada beserta *foreign key*nya.

- Antara entitas dosen dan mata kuliah dengan relasinya adalah mengajar
- Antara dosen dengan admin relasinya adalah registrasi
- Antara mahasiswa dengan mata kuliah relasinya adalah nilai
- Antara mata kuliah dengan jadwal relasinya adalah alokasi
- Antara jadwal dengan ruangan relasinya adalah tempat.

Foreign key adalah *key* pada relasi yang merupakan Primary Key dari entitas entitas yang terhubung oleh relasi tersebut

- Menentukan derajat kardinalitas relasi untuk setiap himpunan relasi
- Relasi mengajar kardinalitasnya N-N
- Relasi registrasi kardinalitasnya 1-1
- Relasi nilai kardinalitasnya N-N
- Relasi alokasi kardinalitasnya 1-N
- Relasi tempat kardinalitasnya 1 -N

Melengkapi entitas dan relasi dengan atribut-atribut deskriptif :

- Menambahkan atribut nama_ dosen pada entitas dosen
- Menambahkan atribut nama_ mhs dan password pada entitas mahasiswa.
- Menambahkan atribut password_ admin pada entitas admin.
- Menambahkan atribut nama_ matkul, SKS_matkul, semester pada entitas mata kuliah.
- Menambahkan atribut nama_ ruangan pada entitas ruangan.

BAB VI

MySQL

Database Management System (DBMS) adalah aplikasi yang dipakai untuk mengelola basis data. DBMS biasanya menawarkan beberapa kemampuan yang terintegrasi seperti:

1. Membuat, menghapus, menambah dan memodifikasi basis data
2. Pada beberapa dbms pengelolaannya berbasis Windows (berbentuk jendela-jendela) sehingga lebih mudah digunakan
3. Tidak semua orang bisa mengakses basis data yang ada sehingga memberikan keamanan bagi data
4. Kemampuan berkomunikasi dengan program aplikasi yang lain. misalnya dimungkinkan untuk mengakses basis data mysql menggunakan aplikasi yang dibuat menggunakan PHP
5. Kemampuan pengaksesan melalui Komunikasi antar komputer (*client server*)

MySQL adalah salah satu aplikasi dbms yang sudah sangat banyak digunakan oleh para pemrogram aplikasi web. contoh DBMS lainnya adalah PostgreSQL (*freeware*), SQL server, MS Access, db2 dari IBM, Oracle dan Oracle Corp, dbase, foxpro, dsb.

Kelebihan dari MySQL adalah gratis, handal, selalu di-*update*, dan banyak forum yang memfasilitasi para pengguna jika memiliki Kendala. MySQL juga menjadi DBMS yang sering di bandling dengan *web server* sehingga proses instalasinya jadi lebih mudah.

6.1 Arsitektur Sistem

Arsitektur sistem maksudnya adalah konfigurasi sistem secara keseluruhan yang menjadi tempat hidup dari dbms, basis data dan aplikasi yang memfaatkannya.

Beberapa jenis arsitektur sistem yang dapat digunakan adalah:

a. Sistem tunggal/ Mandiri (*stand alone*)

Pada sistem ini DBMS, basis data serta aplikasi basis data ditempatkan pada komputer yang sama. arsitektur ini paling sederhana dan murah. Arsitektur ini dapat kita pilih jika basis data yang dikelola tidak terlalu besar dan lebih bersifat untuk membantu pekerjaan administratif. Sistem Mandiri dapat digunakan di rumah masing-masing seperti di apotek kecil, wartel, Hotel kecil, dan sebagainya

b. Sistem tersentralisasi (*centralized system*)

Jika yang disentralisasi adalah DBMSnya, maka servernya disebut *application server*. jika yang disentralisasi adalah basis datanya, maka servernya disebut *file server*. sistem tersentralisasi menggunakan application server kurang baik untuk spesifikasi *server*-yang rendah. karena pekerjaan pada *server* sangat berat. sistem tersentralisasi menggunakan *file server* kurang baik untuk jaringan yang terlalu luas karena transaksi dan data bisa sangat berat dan keamanan kurang terjaga.

c. Sistem *client server*

Arsitektur ini dibuat untuk menutupi kelemahan pada sistem tersentralisasi. Beban *server* jadi tidak terlalu berat

karena klien juga memiliki dbms sehingga proses yang bisa dilakukan di klien akan dilakukan di *client*. lalu lintas data antara *server* dan Workstation pun dibuat lebih efisien.

6.2 Tipe Data String

MySQL menggunakan seluruh tipe data numerik standar ANSI. Berikut ini adalah tipe data numerik yang biasanya digunakan beserta penjelasannya.

Tipe Data	Deskripsi
INT	nilai integer yang bisa bertanda atau tidak. jika bertanda, maka rentang yang diperbolehkan adalah -2147483648 sampai 2147483647, sedangkan jika tidak bertanda maka rentang nya dari 0 sampai 4294967295
TINYINT	nilai integer yang sangat kecil. Rentangnya 128-127 untuk yang bertanda dan 0- 255 Untuk yang tidak bertanda.
SMALLINT	nilai integer yang sangat kecil dengan rentang 31768 sampai 32767 untuk yang bertanda Sedangkan untuk yang tidak bertanda dari 0 sampai 65535.
MEDIUMINT	integer dengan ukuran sedang dengan rentang -8388608 sampai 8388607 atau 0 sampai 16777215

BIGINT	Integer dengan ukuran besar dengan rentang - 922337203685775808 sampai 9223372036854775807 atau 0 sampai 18446744073709551615.
FLOAT(M,D)	bilangan pecahan dengan panjang (termasuk jumlah desimal) M dan jumlah desimal D. presisi desimalnya bisa sampai 24 digit. defaultnya float (10,2). bilangan Float selalu bisa bertanda.
DOUBLE(M,D)	adalah bilangan pecahan dengan presisi dua kali lipat. panjang m dan jumlah desimal d. presisi desimalnya bisa sampai 53 digit. default-nya double (16,4) bilangan float selalu bisa bertanda. sinonim dari double adalah real.
DECIMAL(M,D)	adalah bilangan pecahan dan harus didefinisikan m&d nya. setiap desimal membutuhkan tempat 1 byte. sinonim dari desimal adalah numerik.

6.3 Tipe Data Tanggal Dan Waktu

Berikut ini adalah tipe data tanggal dan waktu didalam MySQL.

Tipe Data	Deskripsi
-----------	-----------

DATE	Adalah tipe data tanggal dengan format YYYY-MM-DD, antara 1000-01-01 dan 9999-12-31. Contoh: 17 Agustus 1945 akan disimpan sebagai 1945-08-17
DATETIME	Adalah kombinasi tanggal dan waktu dengan format YYYY-MM-DD HH:MM:SS dan rentang data antara 1000-01-01 00:00:00 sampai dengan 9999-12-31 23:59:59. Contoh pukul 10:00 pagi pada tanggal 17 Agustus 1945 akan disimpan sebagai 1945-08017 10:00:00.
TIMESTAMP	Sebuah penanda waktu antara 1 Januari 1970 tengah malam sampai dengan tahun 2037. Formatnya mirip dengan DATETIME tetapi tanpa pembatas diantara angkanya. Contoh: pukul 10:00 pagi pada tanggal 27 Agustus 1945 akan disimpan sebagai 19450817100000.
TIME	Menyimpan waktu dalam format HH:MM:SS. Contoh pukul 10:00 akan disimpan menjadi 10:00:00
YEAR(M)	Menyimpan data tahun dalam format 2 atau 4 digit. Jika M diisi dengan nilai 2, maka rentang tahunnya dari 1970-2069 sedangkan jika M diisi dengan

nilai 4 maka YEAR bisa berisi 1901 sampai dengan 2155. Default nilai M adalah 4.

6.4 Tipe Data String

Berikut ini adalah tipe data string yang paling umum di dalam MySQL.

Tipe Data	Deskripsi
CHAR	string dengan ukuran tetap. ukurannya antara 1 sampai 255 karakter. Ukuran ditentukan dengan nilai M. Contoh CHAR(6).
VARCHAR	string dengan ukuran bervariasi antara 1 sampai dengan 255 karakter. Contoh VARCHAR(25)
TEXT	String dengan ukuran maksimum 65535 karakter. String yang tersimpan di dalam teks dianggap tidak <i>case sensitive</i> . Untuk kapasitas yang lebih kecil bisa menggunakan TINYTEXT dengan kapasitas maksimal 255 karakter sedangkan untuk kapasitas yang lebih besar bisa menggunakan MEDIUMTEXT (16777215 karakter) dan LONGTEXT (4294967295 karakter).

BLOB

Binary Large Objects (BLOB) adalah tipe data untuk menyimpan data binary dalam jumlah besar. biasa digunakan untuk menyimpan citra. Untuk penyimpanan data yang lebih kecil bisa menggunakan TINYBLOB (maksimal 255 karakter) Sedangkan untuk kapasitas yang lebih besar bisa menggunakan MEDIUMBLOB (maksimal 16777215 karakter) dan LONGBLOB (maksimal 4294967295 karakter)

ENUM

enumerasi atau sebuah list atau daftar. jadi misalnya anda ingin bahwa sebuah nilai terbatas hanya boleh dengan nilai tertentu saja maka anda bisa membuat sebuah daftar. nilai itu hanya bisa terdiri dari a sampai e maka anda bisa membuatnya menjadi Enum (A,B,C,D,E)

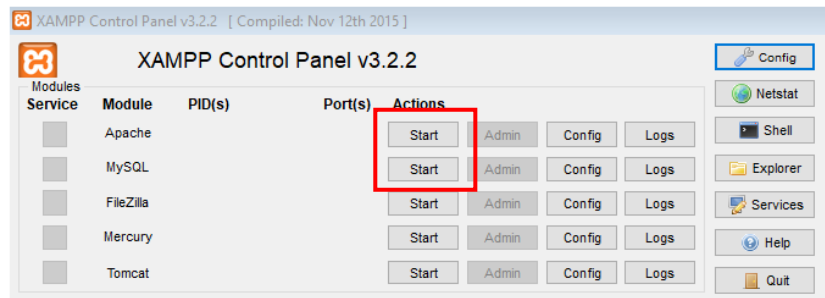
6.5 phpMyAdmin

Secara definisi, phpMyAdmin adalah *tool open source* yang ditulis dalam bahasa PHP untuk menangani administrasi my SQL berbasis *World Wide Web*.

Cara membuka phpmyadmin adalah sebagai berikut:

1. Bukalah XAMPP Control Panel dengan cara klik kanan XAMPP Run di administrator

2. setelah itu tekan Start pada baris Apache, tekan Start pada baris my SQL. kemudian tekan admin pada baris MySQL



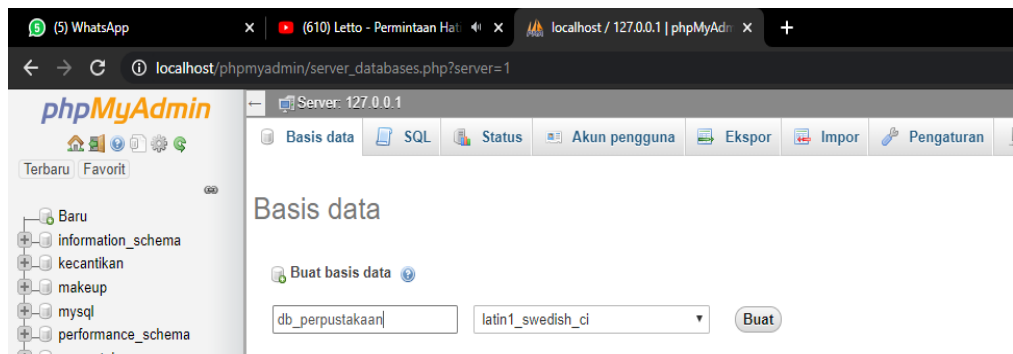
Tools ini digunakan untuk :

- a. Membuat database

Misalnya kita ingin membuat basis data dengan nama db_perpustakaan.

Langkah kerjanya adalah :

- Buka phpMyAdmin, kemudian isikan nama database yaitu db_perpustakaan kemudian tekan Buat.



- b. Membuat Tabel.

Pada dasarnya, data-data dalam basis data disimpan di dalam tabel. untuk merancang tabel, langkah kerjanya adalah:

- Pilih basis data tempat kita membuat tabelnya (db_perpustakaan).
- Isikan mahasiswa pada name dan dua pada number of cplomns kemudian tekan go di sebelah kanan bawah
- Pada kolom pertama diisi dengan (column : id_mahasiswa, type:CHAR, length: 6, index: primary). Isian yang lain Biarkan saja

BAB VII

LANDASAN TEORI WEB SERVICE

7.1 SOA (*Service Oriented Architecture*)

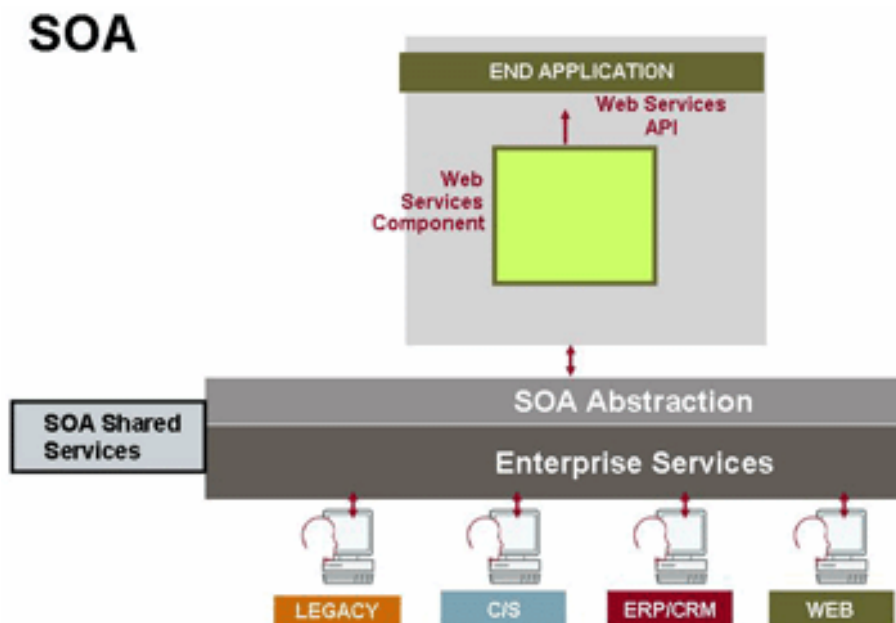
SOA adalah sebuah skema yang memungkinkan komunikasi antar sistem dilakukan secara *loosely coupled*, artinya masing-masing pihak tidak memiliki ketergantungan yang tinggi satu sama lain, dengan kata lain SOA sebuah arsitektur Kerangka kerja berbasis standar terbuka yang memungkinkan perusahaan-perusahaan untuk saling mengintegrasikan data satu sama lain dengan sebuah layanan yang diberikan, Sehingga aplikasi A bisa mendapatkan informasi dari aplikasi B begitu juga sebaliknya.

Menurut Rahmi Nur Shofa, et al. (2013) SOA sendiri merupakan suatu konsep gaya arsitektural yang memodularisasi informasi menjadi *services*, dengan demikian *web service* merupakan teknologi yang tepat untuk menerapkan konsep SOA, karena *cycle* proses *web service* sama dengan SOA. Adapun gambaran dari Soa dapat dilihat pada gambar 7.1.

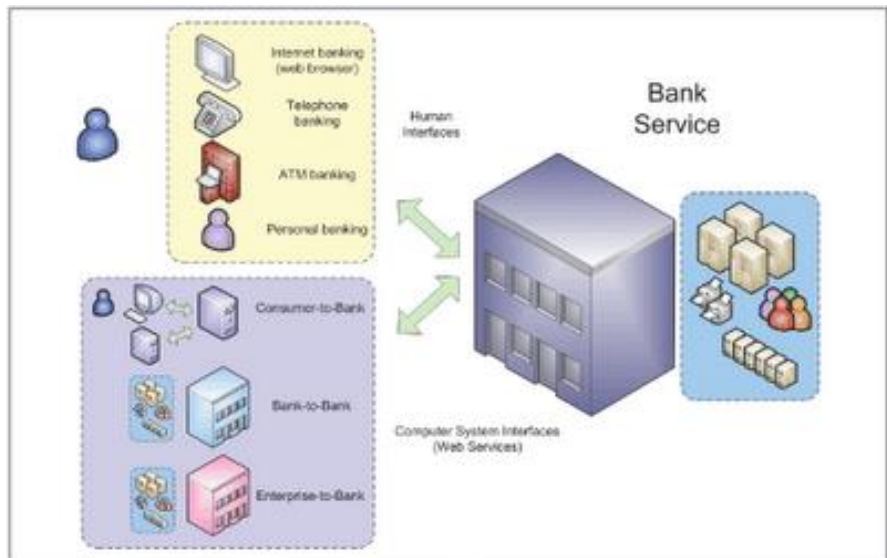
Dari gambar 7.1 arsitektur dapat dilihat SOA menyediakan sebuah layanan yang terdapat pada *web service*, di mana layanan tersebut setelah dibagikan kepada pengguna aplikasi, suatu konsep biasanya dikenal dari karakteristiknya. Dalam SOA terdapat beberapa karakteristik yaitu:

1. *Loose Coupled* adalah *service* yang dipanggil oleh *service* lainnya tanpa program pemanggil tersebut perlu memperhatikan Dimana lokasi servis yang dipanggil berada dan platform apa yang digunakan oleh *service* tersebut.

Loose Coupling sangat penting bagi SOA, karena dengan demikian pemanggilan sebuah *service* oleh *service* lainnya dapat dilakukan pada saat *run-time*. Untuk lebih jelasnya dapat dilihat pada gambar 7.2.

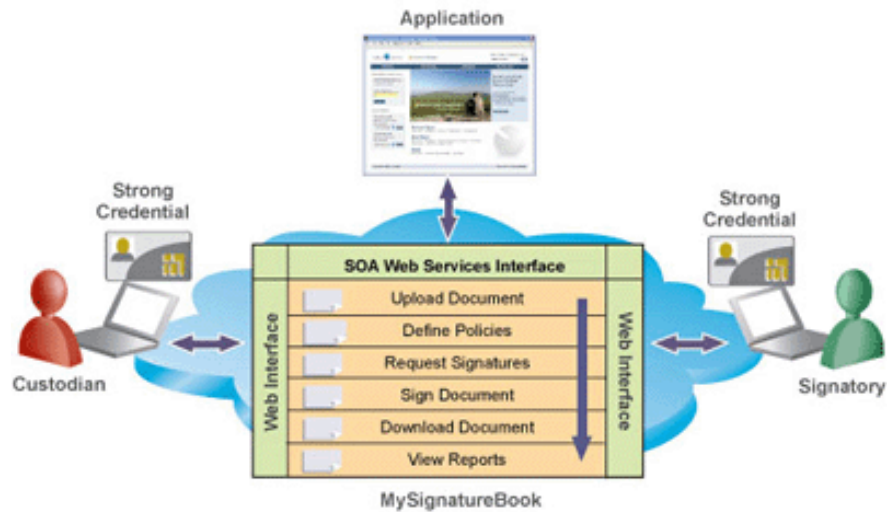


Gambat 7.1 Arsitektur SOA



Gambar 7.2 Loose Coupled

2. Soa tersusun dari *service interface* dan *service implementation*. *Service interface* menyatakan Bagaimana parameter dari *input* dan *output* serta lokasinya berada, sedangkan *service implementation* adalah *logic* proses dari sebuah *service*. Adapun arsitektur dari *service interface* dan *service implementation* dapat dilihat pada gambar 7.3



Gambar 7.3

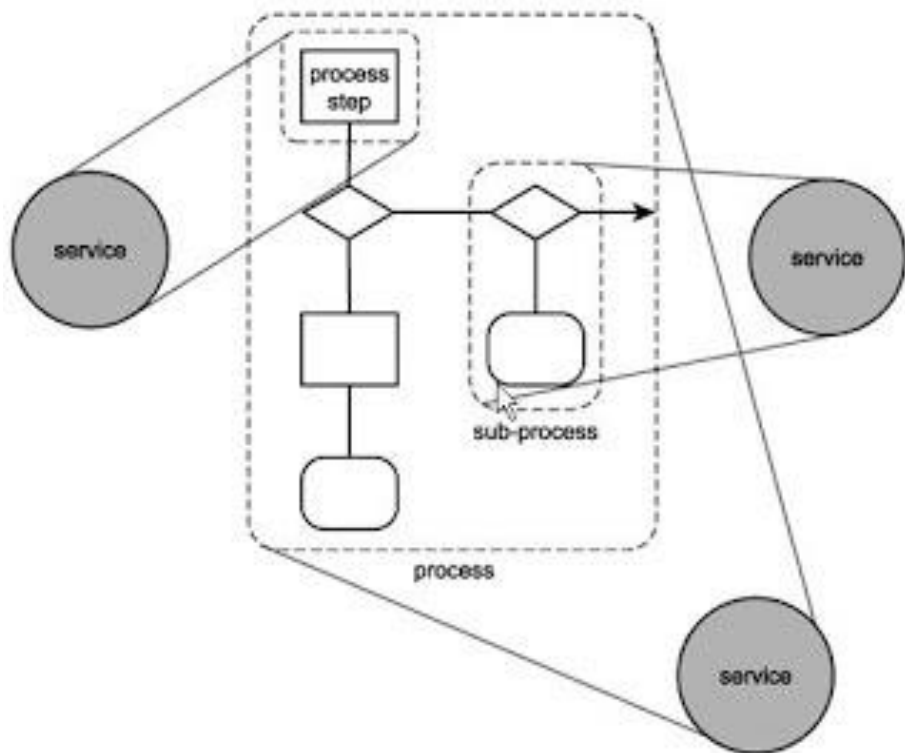
Dari gambar 7.3 terlihat Bagaimana skema yang akan dibangun pada sistem informasi pembayaran uang kuliah online, aplikasi yang saling berkomunikasi antar beda *platform*, dari rancangan tersebut perlu adanya sebuah rancangan yang mengubah bisnis *logic* menjadi sebuah layanan yang dibungkus dalam sebuah wadah aplikasi jawabannya adalah *web service*.

3. *Service* harus *business oriented* adalah setiap *service* yang disediakan harus melakukan suatu aktivitas bisnis tertentu. Adapun Bagaimana enkapsulasi rojik dalam servis dapat dilihat pada gambar 7.4

Dari gambar 2.4 dapat dilihat Bagaimana sebuah servis kapsulasi sebuah bisnis *Logic* menjadi *service-service* kecil.

Manfaat utama dari SOA adalah *service oriented* atau berorientasi pada layanan untuk memungkinkan institusi Dalam

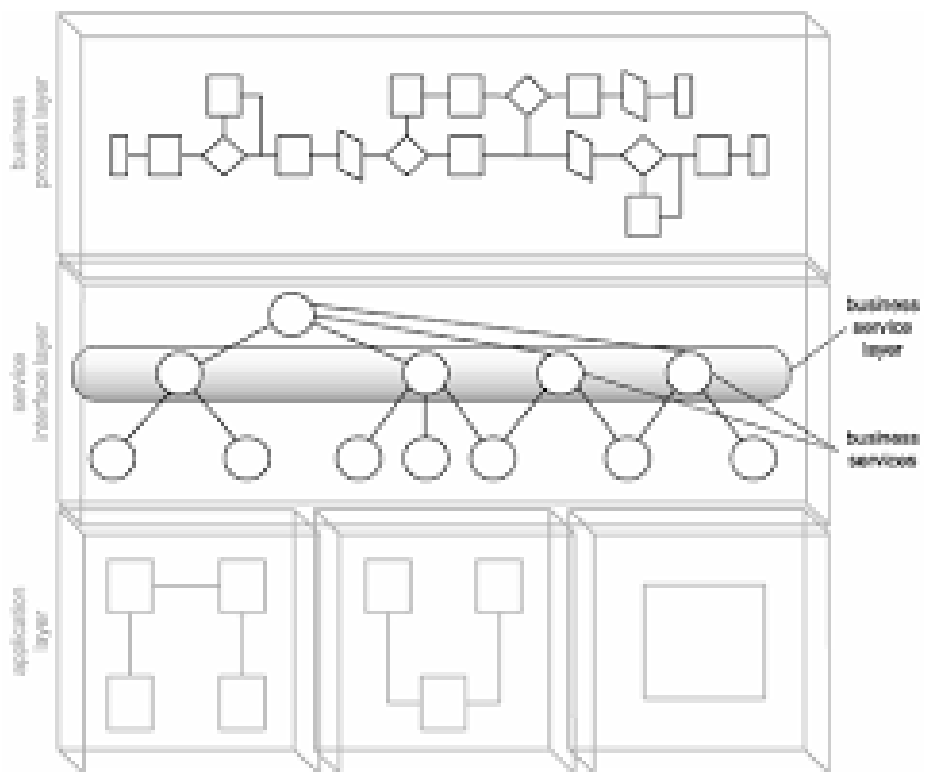
penggunaannya secara simultan dan mudah untuk pertukaran data antar program dari *vendor* yang berbeda dengan menggunakan sebuah layanan, dan juga antar institusi tidak perlu menggunakan *platform* yang sama dalam pemrograman maupun database sehingga SOA menjadikan sebuah solusi bagi institusi-institusi untuk memberikan inovasi layanan tanpa memerlukan biaya yang tinggi dalam pencapaian inovasi tersebut.



Gambar 7.4 Enkapsulasi logic Dalam Service

Riyanarto Sarno (2012) dalam sebuah pengembangan perangkat lunak menggunakan pendekatan konsep SOA berbeda

dengan pengembangan tanpa menggunakan konsep SOA, di mana sebuah perangkat lunak yang dikembangkan tidak menggunakan SOA mengakibatkan aplikasi berjalan pada dua *layer* utama yaitu *application layer* dan *business process layer*, sementara itu dalam menggunakan konsep SOA diimplementasikan dalam sebuah *layer* di antara *application layer* dan *business process layer* yaitu *service interface layer*, fungsi ini adalah menggambarkan *application logic* dan *business process* yang ada di *business logic*. Lihat gamabr 7.5



Gambar 7.5 Layering Dalam SOA

Gambar 7.5 terlihat bahwa teknologi aplikasi a b dan c terenkapsulasi oleh *service interface layer*, terlihat bahwa application logik dan bisnis logic dibungkus dalam suatu layanan. dan *service interface layer* itu sendiri memiliki tiga lapisan abstraksi yaitu

1. *Application service layer*

Sebuah layanan yang letaknya berbatasan dengan *application layer* yang bertugas sebagai penyedia fungsi-fungsi untuk pemrosesan data yang nantinya akan dibutuhkan untuk menjalankan fungsi-fungsi yang ada pada proses bisnis.

2. *Business service layer*

Digunakan untuk mengimplementasikan *logic* dan model bisnis menjadi sebuah *service*.

3. *Orchestration service layer*

Sebuah layanan yang memberikan abstraksi Bagaimana *service-service* yang berjalan sesuai dengan aturan dan urutannya yang sesuai dengan logika dan bisnis proses yang ada pada perusahaan.

Dengan menggunakan pendekatan metode SOA, langkah dalam pembuatan sistem informasi pembayaran uang kuliah online dapat lebih mudah dipahami, sehingga kita dapat mengetahui bagaimana pengembangan sistem informasi dapat terbentuk, yang akan menjadi sebuah acuan kerja dalam pengembangan sistem informasi secara teknis.

“Arif Firmansyah (2011), dalam dunia perbankan sebenarnya setelah ada arsitektur bisnis yang mirip dengan SOA.

Bank besar di dunia telah diimplementasikannya untuk sistem komunikasi data transaksi *point of sale* (POS) untuk pembayaran kartu kredit di tokoh-tokoh dan auto *teller machine* untuk proses transfer pembayaran dan penarikan tunai.”

Dengan demikian, perancangan sebuah teknologi *web service* dalam pembayaran uang kuliah online ini hendaknya dapat memberikan kontribusi yang baik bagi bank, begitu juga dengan pihak kampus

7.2 Tentang Web Service

Web Service merupakan suatu komponen *software* yang dirancang untuk mendukung interaksi antar sistem pada suatu jaringan, *web service* digunakan sebagai suatu fasilitas yang disediakan oleh suatu *web site* untuk menyediakan layanan dalam bentuk informasi kepada sistem lain, sehingga sistem lain dapat berkomunikasi dengan sistem tersebut melalui layanan layanan atau *service* yang disediakan oleh suatu sistem yang menyediakan *web service*.

“Utomo Budiyanto dan Habib Musthofa (2014), *web service* adalah sebuah antarmuka yang terletak di antara kode aplikasi dan pengguna kode tersebut, yang berperan sebagai lapisan abstrak yang memisahkan *platform* dan rincian spesifik bahasa pemrograman tentang bagaimana kode aplikasi seharusnya dipanggil”.

Web service merupakan komponen perangkat lunak yang dirancang untuk mendukung interaksi mesin ke mesin melalui sebuah jaringan melalui HTTP. Memungkinkan sebuah aplikasi untuk berkomunikasi jarak jauh melalui *internet* atau *intranet* secara independen menggunakan sebuah *platform* dari sebuah bahasa pemrograman menggunakan aturan pertukaran pesan berbasis XML.

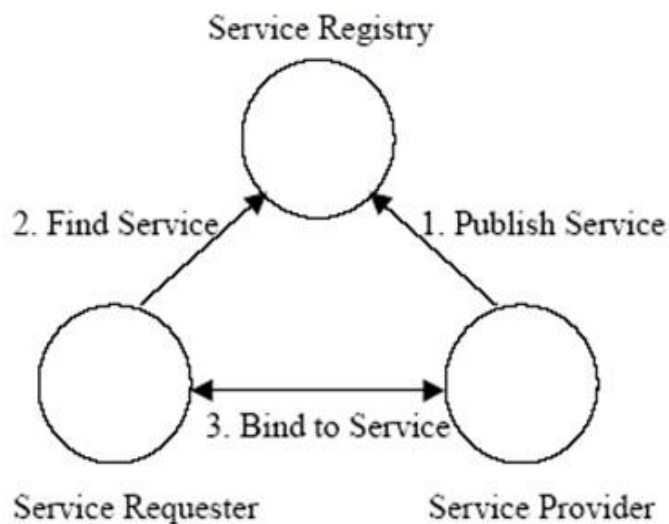
“Menurut Edi Sultanta dan Khabib Musthofa (2013), *Web service* dapat dipahami sebagai *remote procedure call* (RPC) yang mampu memproses fungsi-fungsi yang didefinisikan pada sebuah aplikasi web serta mengekspos sebuah *user interface* melalui *web* dan juga sebagai jembatan penghubung dengan database tanpa perlu *driver database* dan tidak harus mengetahui jenis DBMS”.

Berbeda dengan aplikasi *web* konvensional, *web service* tidak mengembalikan *interface* yang dapat langsung digunakan oleh *user* berupa layanan yang berbentuk kode HTML, tetapi *web service* mengembalikan layanan tersebut berupa kode XML yang berorientasi pada data yang langsung dapat digunakan oleh *user*.

Keunggulan teknologi *web service* sebenarnya menjadi suatu solusi yang tepat bagi pengembang sebuah sistem informasi, karena penggunaan *internet* yang sudah merasuk ke semua bidang termasuk dunia bisnis dan industri, *web service* menjawab akan tuntutan adanya aplikasi *business to business*, *application to application* yang memungkinkan untuk sebuah layanan *internet* yang dapat digunakan oleh aplikasi lain, karena *web service* dapat menjembatani sebuah aplikasi yang berbeda

framework sekalipun berbeda dalam pemilihan *database* yang digunakan

Operasi suatu *web service* dapat digambarkan dengan sebuah arsitektur yang memiliki tiga komponen yang mempunyai peranan berbeda-beda. Lihat gambar 7.6



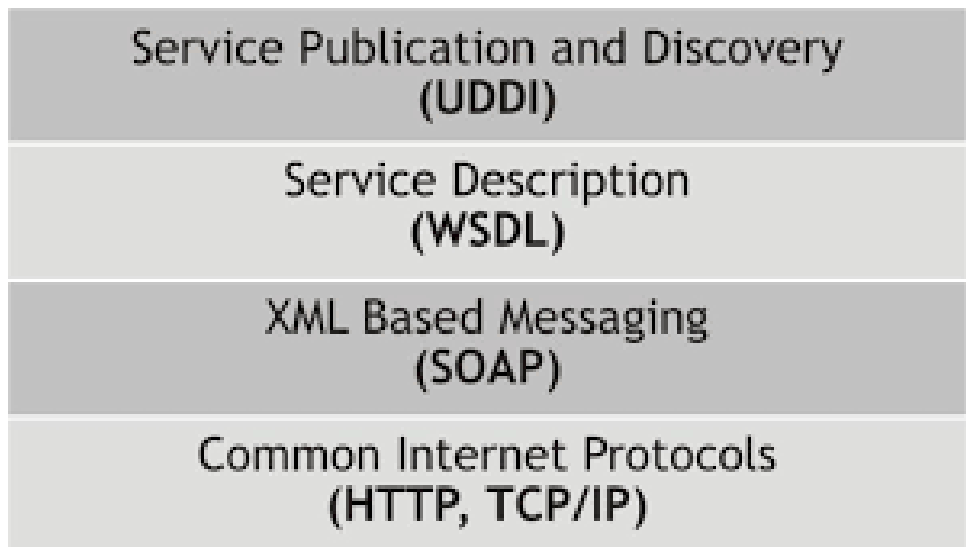
Gambar 7.6 Arsitektur Web Service

Keterangan Gambar 7.6

- *Service provider* atau Penyedia Layanan
Berfungsi sebagai lokasi Sentral yang mendeskripsikan semua layanan atau service yang sudah deregister
- *Service Registry* atau Daftar Layanan
Meminta layanan yang mencari dan menemukan layanan yang dibutuhkan oleh *service requestor*, serta menggunakan layanan tersebut
- *Service Requestor* atau Peminta Layanan

Adalah pihak yang membutuhkan sebuah layanan atau service dan mengolah sebuah registry agar layanan-layanan tersebut dapat tersedia

Secara keseluruhan komponen-komponen dari web service dapat dilihat pada gambar 7.6



Gambar 7.6 Blok Bangunan Web Service

Web Service secara keseluruhan mempunyai empat komponen seperti pada gambar 6.7 yaitu:

1. Layar 1: protokol *internet* standar yang digunakan sebagai media transportasi adalah http, TCP/IP.
2. Layar 2: *Simple Object Access Protocol* (SOAP), merupakan protokol akses objek berbasis XML yang digunakan untuk pertukaran data antar layanan.
3. Layar 3: *Web Service Definition Language* (WSDL), merupakan suatu standar bahasa dalam format XML yang

berfungsi untuk mendeskripsikan seluruh layanan yang tersedia.

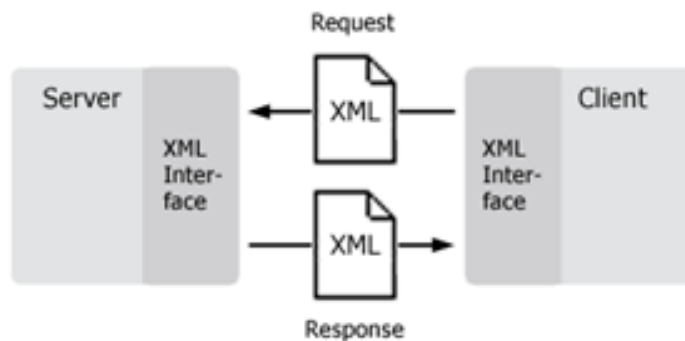
4. Layar 4: *Universal Description Discovery and Integration* (UUDI), yang mana merupakan direktori pusat untuk deskripsi layanan.

“Menurut Riyanto (2013), UDDI merupakan suatu layanan direktori hierarki dan terpusat yang menyediakan layanan untuk mempublikasikan informasi teknis layanan web.

Dari beberapa penjabaran mengenai *web site* yang telah kita bahas di atas dapat disimpulkan beberapa kelebihan *web service* diantaranya:

- *Web service* mempunyai kelebihan untuk bisa diakses oleh aplikasi yang berjalan pada *platform* yang berbeda-beda.
- *Web service* menggunakan standar dan protokol terbuka pada internet dengan menggunakan HTTP.
- *Web Service* memungkinkan fungsi-fungsi pada banyak perangkat lunak di *internet* untuk dipadukan menjadi *web service* baru.

Disamping itu, web service mempunyai prinsip kerja yaitu mengembalikan hasil servis ke dalam bentuk XML yang berorientasi pada data yang belum bisa dibaca oleh pengguna, sehingga untuk menampilkan data tersebut diperlukan sebuah *interface*. Lihat Gambar 7.8



Request/response scheme (Client → Server → Client)

Gambar 7.8 Pengiriman Data XML

Application pada gambar 7.8 berfungsi untuk menampilkan data XML ke *database* bentuk *interface*, sehingga data dalam *interface* tersebut diberikan kepada pengguna dalam bentuk *user interface*.

Untuk merepresentasikan kontrak antara *requestor* dan *provider* atau antara kode *client* dan kode *server*, secara teknis dapat digunakan beberapa standar protokol yang sudah ada yaitu SOAP WSDL. Dengan menggunakan SOAP WSDL, *client* dapat memanfaatkan fungsi-fungsi *public* yang disediakan oleh *server*.

WSDL (*web service description language*) merupakan sebuah dokumen dalam format XML yang isinya menjelaskan informasi detail sebuah *web service*. Di dalam WSDL di jelaskan metode-metode apa saja yang tersedia, parameter apa saja yang diperlukan untuk memanggil sebuah metode, apa hasil atau tipe data yang dikembalikan oleh metode yang dipanggil tersebut.

Dalam sebuah pengembangan dan pembuatan sistem informasi dibutuhkan sebuah perancangan yang baik agar sistem informasi yang dikembangkan dapat berjalan sesuai dengan yang

dibutuhkan sehingga sistem tersebut menjadi konsisten, terarah sesuai dengan aturan, terkonsep dan dapat dikembangkan dengan mudah di kemudian harinya.

7.3 XML (*Extensible Markup Language*)

Xml merupakan sekumpulan aturan-aturan yang mendefinisikan suatu syntax yang digunakan untuk menjalankan suatu teks atau data dalam sebuah dokumen melalui penggunaan *tag*.

Di dalam *web service*, XML berfungsi sebagai komunikasi antar aplikasi dan integrasi data walaupun antar aplikasi yang berkomunikasi tersebut berbeda *platform*. Di sini juga terletak kelebihan dari XML tersebut, aplikasi-aplikasi yang berbeda dapat dengan mudah berkomunikasi antar satu dengan yang lain.

Di dalam jurnal implementasi *extensible markup language web service*, XML *web service* memiliki beberapa karakteristik diantaranya:

- Standar dari XML *web service* adalah Standar Industri yang independen.
- XML *web service* tidak menyediakan *interface*, tetapi hanya menyediakan fungsi.
- XML *web service* menggunakan *hypertext Transfer Protocol* (HTTP) sebagai protokol standar.
- XML *web service* menggunakan model *request-response* yang sama seperti pada aplikasi *web*.

- XML *web service* ditempatkan di dalam sebuah *web server*.
- XML *web service* lebih ditujukan untuk kepentingan dari *programmer to programmer*, sehingga penggunaan XML *web service* yang utama bukanlah pengguna akhir tapi *programmer* yang menggunakan bahasa pemrograman yang berbeda.
- Hasil dari XML *web service* merupakan *plain text* yang berupa berkas XML.
- Fungsi dari sebuah XML *web service* butuh pengolahan lebih lanjut dalam membentuk sebuah *interface* aplikasi.
- Secara umum hampir semua bahasa pemrograman yang dapat terkoneksi ke *internet* akan mampu terkoneksi dengan XML *web service*, namun tidak semua bahasa pemrograman tersebut mampu memproduksi XML *web service*.

Pada dasarnya, XML tidak jauh berbeda dengan HTML, dimana elemen yang digunakan sama-sama menggunakan pembuka dan penutup, masing-masing dikembangkan untuk tujuan yang berbeda-beda. HTML digunakan sebagai pemberi informasi yang dapat terlihat, sedangkan XML mendeskripsikan susunan informasi dan berfokus pada informasi itu sendiri.

7.4 SOA (*Simple Object Access Protocol*)

SOAP merupakan protokol untuk pertukaran informasi dengan desentralisasi dan terdistribusi. SOAP merupakan gabungan antara HTTP dengan XML, karena umumnya

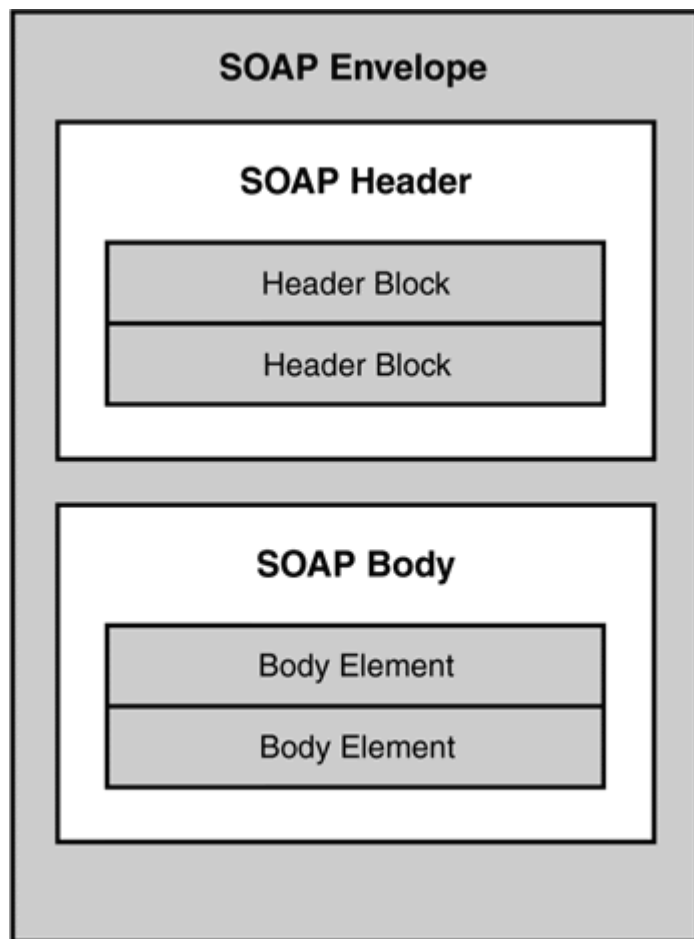
menggunakan protokol HTTP sebagai sarana *transport* datanya dan data akan dipertukarkan atau ditulis dalam format XML.

Karena SOAP menggunakan HTTP dan XML maka memungkinkan pihak-pihak yang mempunyai *platform*, seperti sistem operasi dan perangkat lunak yang berbeda dapat saling mempertukarkan datanya. SOAP mengatur bagaimana *request* dan respon dari suatu *web service* bekerja. (Hartati Deviana, 2013).

“Riyanto,*et al.* (2013) pesan SOAP tersusun dari sebuah amplop yang berisi tubuh pesan dan informasi *header* yang akan dipakai untuk menerapkan pesan tersebut. elemen *envelope*. Dalam amplop tersebut, pesan SOAP yang sah dapat berisi elemen-elemen anak lainnya”. Adapun struktur pesan SOAP sebagai berikut:

- *Envelope Element* (yang mengidentifikasi dokumen XML sebagai pesan SOAP).
- *Elemen Header* (berisi informasi *header*).
- *Elemen body* (berisi panggilan dan respon informasi)

Jika digambarkan dalam sebuah diagram maka akan terlihat seperti pada gambar berikut.



Gambar 2.10 Diagram Struktur SOAP

7.5 WSDL (*Web Service Description Language*)

WSDL merupakan bahasa berbasis XML yang digunakan untuk mendefinisikan *service* dan menggambarkan Bagaimana cara mengakses *web service* tersebut.

Ketika aplikasi *client* meminta *service*, UDDI akan memberikan informasi tentang tata letak dari dokumen WSDL, WSDL berisi sebuah pesan dengan skenario XML dengan skema tersebut pesan yang diminta dari *client* akan diproses dengan menggunakan WSDL, *client* dapat memanfaatkan fungsi-fungsi

publik yang sudah disediakan oleh *server* berupa *method-method* dan parameter-parameter apa saja yang tersedia dalam sebuah *web service*.

“Menurut Danny sasmoko (2013), WSDL adalah dokumen XML yang *machine readble*. Oleh karena itu, dokumen digunakan untuk melakukan otomatisasi prosesPengintegrasian layanan ke aplikasi *requestor*, aplikasi *requestor* tersebut langsung berhubungan dengan *service provider*.

Elemen-elemen yang ada pada WSDL adalah:

- *Message*

Definisi tipe data yang akan dikomunikasikan.

Berikut contoh struktur kode message WSDL:

```
34 <message name="searchRequest">
35   <partname="param" type="tns:TypeDataInput"/></message>
36
37 <message name="searchResponse">
38   <partname="return" type="tns:TypeDataOupptut"/></message>
39
```

Pada contoh *script message Element* WSDL di atas diperlihatkan tersebut adalah sekumpulan argumen yang di dalamnya terdapat sejumlah elemen pesan yang digunakan sebagai *input output* atau *fault message* dalam suatu operasi

- *Operation*

Nama operasi dari sebuah *web site* atau *method* untuk proses *input output*.

Berikut contoh struktur *coding Operation* WSDL:

```
41 <operation name="search">
42   <documentation>Pemba Uang Kuliah</documentation>
43   <input message="tns:searchRequest"/>
44   <output message="tns:searchResponse"/>
45 </operation>
```

Contoh *script Operation element* WSDL di atas diperlihatkan di mana *pers* merupakan nama *method* yang akan digunakan pada proses *request* data bagi *client* ke *server*, penamaan ini tidak menjadi sebuah ketentuan bagi operasi, kita dapat memberikan nama sesuai dengan kebutuhan kita.

Documentation merupakan sebab penamaan atau *header* dari sebuah pengiriman *file*.

Input merupakan operasi masukan bagi parameter yang dikirimkan.

Output merupakan operasi keluaran atau *response* yang diberikan kepada *server* kepada *client*.

- *Port Type*

Mendeskripsikan sebuah *web service*, operasi-operasi yang telah dijalankan dan pesan-pesan sebuah *web service*.

Berikut contoh struktur kode *port tupe* WSDL:

```
8
9 <portType name="serverportType">
10   <operation name="search">
11     <documentation>Pmebayaran Uang Kuliah</documentation>
12     <input message="tns:searchRequest"/>
13     <output message="tns:serachResponse"/>
14   </operation>
15 </portType>
16
```

Pada contoh *script code port type* elemen WSDL di atas penjelasannya persis dengan penjelasan pada *operation*, namun yang menjadi di tambah adalah penamaan *port type* yang digunakan

- *Binding*

Protokol komunikasi yang digunakan oleh *web service*.

Berikut contoh struktur kode dinding WSDL:

```
17
18 <binding name="serverBinding" type="tns:unitasserverPorttype">
19   <soap:binding style="rpc"
20     transport="http://schemas.xmlsoap.org/soap/http"/>
21
22   <operation name="search">
23     <soap:operation
24       soapAction="urn:server#search" style="rpc"/>
25     <input><soap:body use="encoded" namespace="urn:server"
26       encodingStyle="http://schemas.xmlsoap.org/soap/encoding"/>
27     </input>
28     <output><soap:body use="encoded" namespace="urn:server"
29       encodingStyle="http://schemas.xmlsoap.org/soap/encoding"/>
30     </output>
31   </operation>
32 </binding>
```

Pada contoh *script code billing* elemen WSDL di atas adalah menentukan pada *web service request* tentang bagaimana membuat pesan anak untuk protokol tertentu, setiap *port type* dapat memiliki satu atau lebih elemen yang terkait. Untuk *port type* yang telah ditentukan, elemen yang mengikat harus menentukan pesan dan pasangan *transportnya* (SOAP / HTTP / SOAP / SMTP).

- *Port*

Port merupakan definisi dari sebuah alamat jaringan..

Berikut contoh struktur *code port* WSDL:

```

1 <port name="serverPort" binding="tns:serverPort">
2 <soap:address location=
3 "http://localhost/pembayarankuliah/trxinputquiry.php"/>
4 </port>

```

Pada contoh *script code port* elemen WSDL diatas diperlihatkan setelah *binding* ditentukan, maka pesan SOAP akan mengarahkan ke alamat mana, nantinya *client* harus masuk untuk menyisipkan parameter *input*.

- *Service*

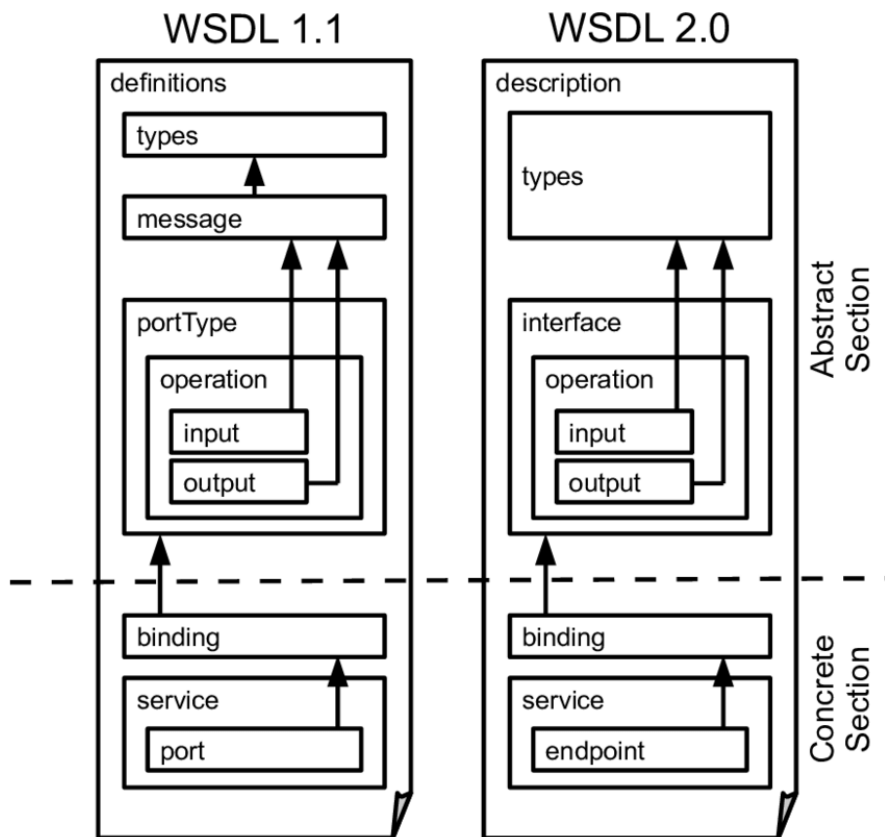
Sekumpulan *endpoint* yang saling berhubungan yang menunjukkan sebuah *file / path* mana yang akan ditempatkan pada file WSDL, setiap penamaan *service* harus diberikan penamaan unik.

Berikut contoh struktur *code service* WSDL:

```

1 <service name="server">
2   <port name="serverPort" binding="tns:serverBinding">
3     <soap:address location=
4       "http://localhost/pembayaranuangkuliah/trxnquiry.php"/>
5   </port>
6 </service>

```

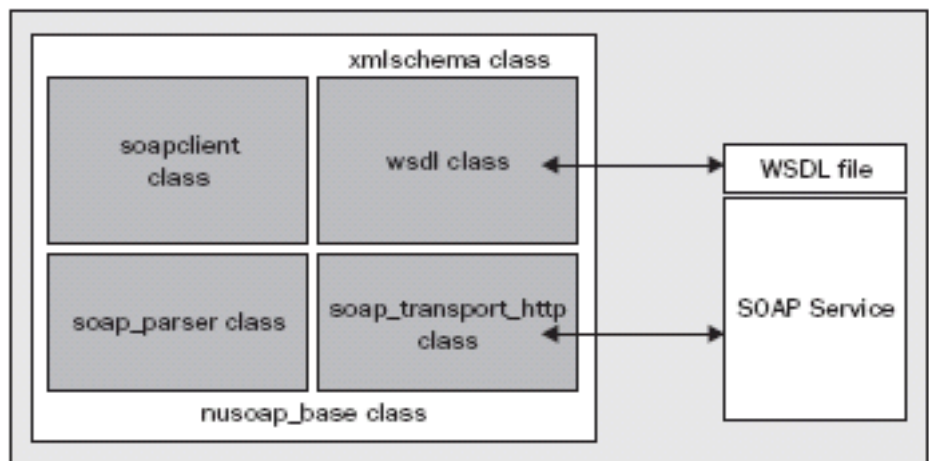
Gambar 2.11 Diagram WSDL

7.6 NuSOAP

NuSOAP merupakan sebuah *library open source* dibawah lisensi NULGPL yang didistribusikan oleh NuSphere Corporation yang berisikan *class PHP* yang memungkinkan *user* untuk mengirim dan menerima pesan SOAP melalui *class client* yang disebut dengan *class “soapclient”* dan *class server* yang disebut dengan *class “soap server”* dan dengan beberapa *class* pendukung lainnya melalui protocol HTTP.

Class “*soapclient*” memanggil data WSDL dari sebuah *file* PHP untuk menerjemahkan parameter-parameternya sekaligus menyusun SOAP *envelope*, ketika pemanggilan tersebut dieksekusi maka class “*soapclient*” menggunakan “*soap_transport_http*” untuk mengirim pesan SOAP *request* lalu pesan SOAP *response* yang diterima di parsing dengan menggunakan “*soap parser*”.

Operasi-operasi pengiriman pesan SOAP dari *client* melalui *method call* () yang mana *method* tersebut berisikan parameter-parameter dari *client* yang akan dibutuhkan untuk menggali sebuah informasi dari *server*. Berikut bentuk diagram proses *web service* menggunakan NuSOAP. Lihat gambar 2.12



Gambar 2.12 Proses web Service Dengan NuSoap

NuSOAP ini juga adalah SOAP Toolkit untuk PHP yang tidak memerlukan ekstensi PHP. Setelah mengunduh NuSOAP, kita perlu mengekstrak konten file zip ke direktori penyimpanan. Setelah diekstraksi, Anda akan mereferensikan “lib / nusoap.php” di aplikasi SOAP PHP Anda.

Contoh :

```
48
49 <?php
50
51 //require NuSOAP
52 require_once("./lib/nusoap.php");
53
54 //retrieve WSDL
55 $client = new nusoap_client("http://{site_url}/service/v4/soap.php?wsdl", 'wsdl');
56
57
```

BAB VIII

ANALISI DAN PERANCANGAN

8.1 Analisis Sistem

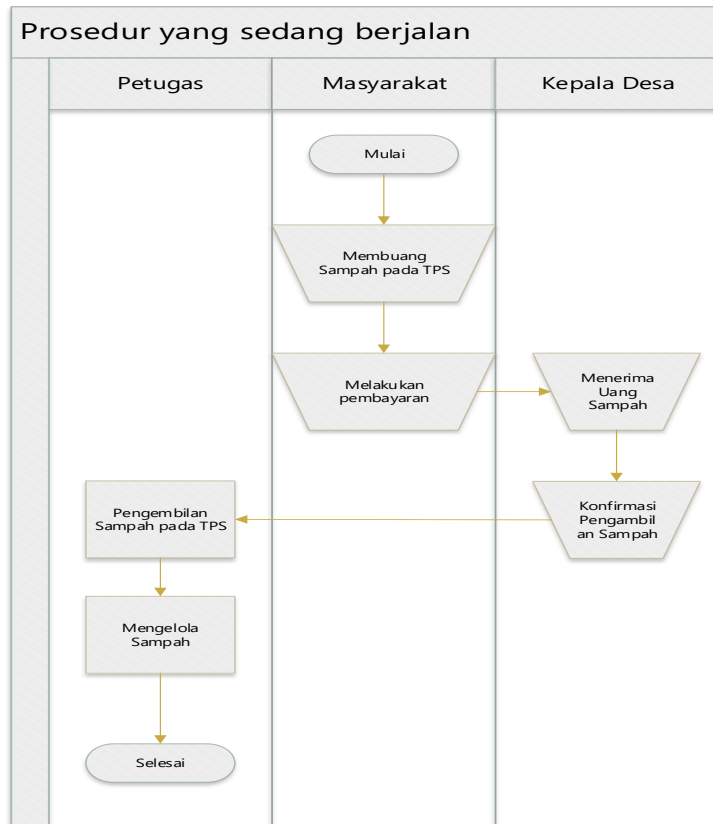
Analisis merupakan langkah awal untuk pengembangan sebuah aplikasi, karena perancangan dan bahkan pengembangan implementasi aplikasi tidak akan berjalan dengan baik tanpa adanya analisa terhadap aplikasi yang akan digunakan. Analisis juga dapat didefinisikan sebagai penguraian dari suatu sistem informasi yang utuh kedalam bagian-bagian komponennya dengan maksud untuk mengidentifikasi dan mengevaluasi masalah-masalah, kesempatan-kesempatan, hambatan-hambatan yang terjadi serta kebutuhan yang diharapkan sehingga dapat diusulkan perbaikan agar mendapat hasil yang maksimal.

Analisis yang dilakukan terhadap Aplikasi Bank Sampah Menggunakan CodeIgniter ini dibuat menggunakan flowmap dan metode Object Oriented yang memberikan gambaran mengenai proses yang terdapat di dalam aplikasi tersebut.

Sistem ini dibangun dengan menggunakan model MVC atau Model View Controller, dimana MVC merupakan model pengembangan dari Object Oriented Programming, dimana setiap baris kodenya dipisahkan menjadi tiga bagian, yaitu ada pada view sebagai form, lalu controller untuk menyimpan fungsi dan class dan model untuk menyimpan database

8.1.1 Analisis sistem yang sedang berjalan

Sistem yang berjalan saat ini terdiri dari satu prosedur yaitu proses di kelolanya sampah oleh kepala desa dengan bantuan petugas kebersihan.



Gambar 8.1 Prosedur yang sedang berjalan

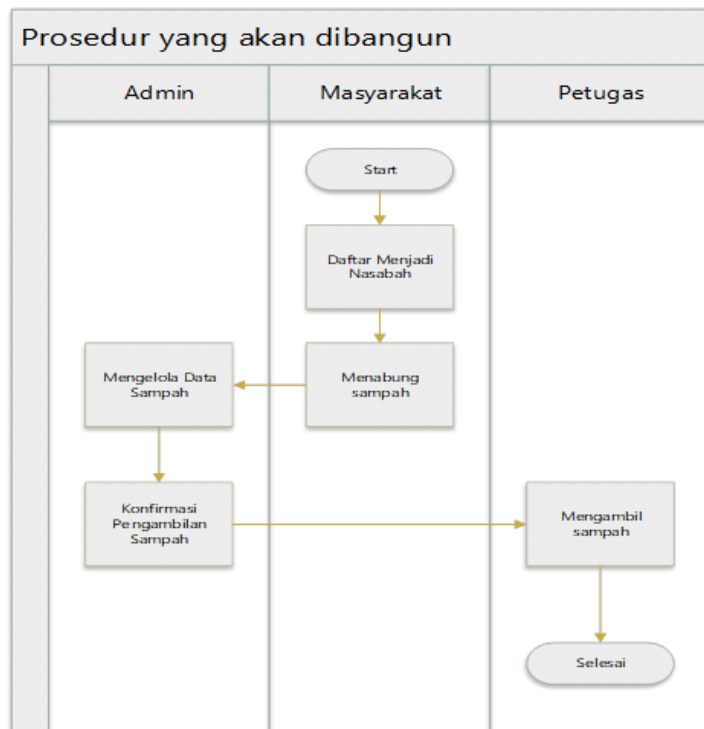
Keterangan :

1. Masyarakat membuang sampah pada TPS yang sudah tersedia di depan rumahnya masing-masing
2. Masyarakat Melakukan pembayaran ke kepala desa setiap sebulan sekali

3. Kepala desa mengkonfirmasi ke pada petugas kebersihan untuk melakukan pengambilan sampah setiap seminggu 2 kali.

8.1.2 Analisis Sistem yang akan di bangun

Pada analisis sistem yang akan dibangun ini, dibuat beberpa pembaruan dari yang sebelumnya. Pada prosedur ini masyarakat di haruskan mendaftarkan terlebih dahulu untuk menjadi nasabah pada Bank Sampah. Prosedur yang akan dibangun pada Bank Sampah yaitu sebagai berikut :



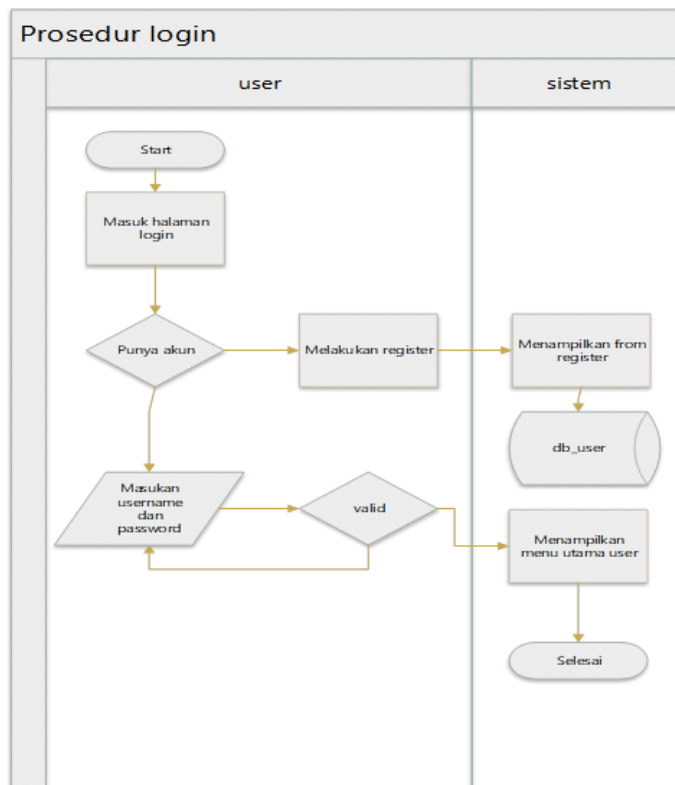
Gambar 8.2 Prosedur yang akan dibangun

Keterangan :

- 1. Masyarakat harus mendaftarkan diri terlebih dahulu untuk menjadi nasabah pada bank sampah**
- 2. Setelah menjadi nasabah, masyarakat dapat menabung sampah pada bank sampah**
- 3. Jika masyarakat menabung sampah, maka data-data sampah tersebut akan di kelola oleh admin**

8.1.3 Analisis Sistem Login yang akan di bangun

Pada analisis sistem yang akan dibangun ini, dibuat beberapa pembaruan dari yang sebelumnya. Pada prosedur ini masyarakat di haruskan mendaftarkan terlebih dahulu untuk menjadi nasabah pada Bank Sampah. Prosedur login yang akan dibangun pada Bank Sampah yaitu sebagai berikut



Gambar 8.3 Prosedur Login

Keterangan :

1. User akan masuk ke halaman login
2. Jika sudah memiliki akun maka user akan langsung melakukan login dan masuk ke halaman user
3. Jika belum memiliki akun maka user akan melakukan registrasi terlebih dahulu
4. User disini yaitu nasabah atau petugas

8.1.3.1 Kebutuhan Fungsional (*Functional Requirements*)

Kebutuhan fungsional adalah jenis kebutuhan yang berisi proses-proses apa saja yang nantinya dilakukan oleh sistem. Kebutuhan fungsional juga berisi informasi-informasi apa saja yang harus ada dan dihasilkan oleh sistem.

Adapun kebutuhan fungsional yang akan dibuat yaitu:

1. Login
2. Kelola data user
3. Kelola data sampah
4. Menabung sampah
5. Pengambilan sampah

Setiap proses memiliki representasi masing-masing pada sebuah tabel atau data yang terdapat pada database yang telah dirancang sebelumnya. Dan setiap proses berhubungan langsung dengan entitas atau user.

8.1.3.2 Kebutuhan Non-Fungsional (*Non-Functional Requirement*)

Analisis kebutuhan non fungsional dilakukan untuk mengetahui spesifikasi kebutuhan untuk sistem. Spesifikasi kebutuhan melibatkan analisis perangkat keras/hardware, analisis perangkat lunak/software, analisis pengguna/user.

Adapun kebutuhan non fungsional yang akan dibuat adalah sebagai berikut :

A. Kebutuhan Perangkat Keras

Pembuatan aplikasi ini menggunakan perangkat sebagai berikut :

Tabel 3.1 Kebutuhan perangkat keras\

No.	Jenis		Keterangan
1	Processor	:	Intel® core™i3
2	Memory	:	4 GB
3	Monitor	:	LCD 14,1 Inchi
4	Mouse dan keyboard	:	Standard

B. Kebutuhan Perangkat Lunak

Spesifikasi perangkat lunak yang digunakan adalah sebagai berikut :

Tabel 3. 2 Kebutuhan perangkat lunak

No.	Jenis	Keterangan
1.	Sistem Operasi	Windows 10 Pro 64-Bit
2.	Server <i>Database</i>	Xampp 1.8.1
3.	Bahasa Pemrograman	PHP dan Android
4.	Software Pendukung	Visual Studio Code dan Android Studio
5.	Browser	Google Chrome

C. Analisis Pengguna/User

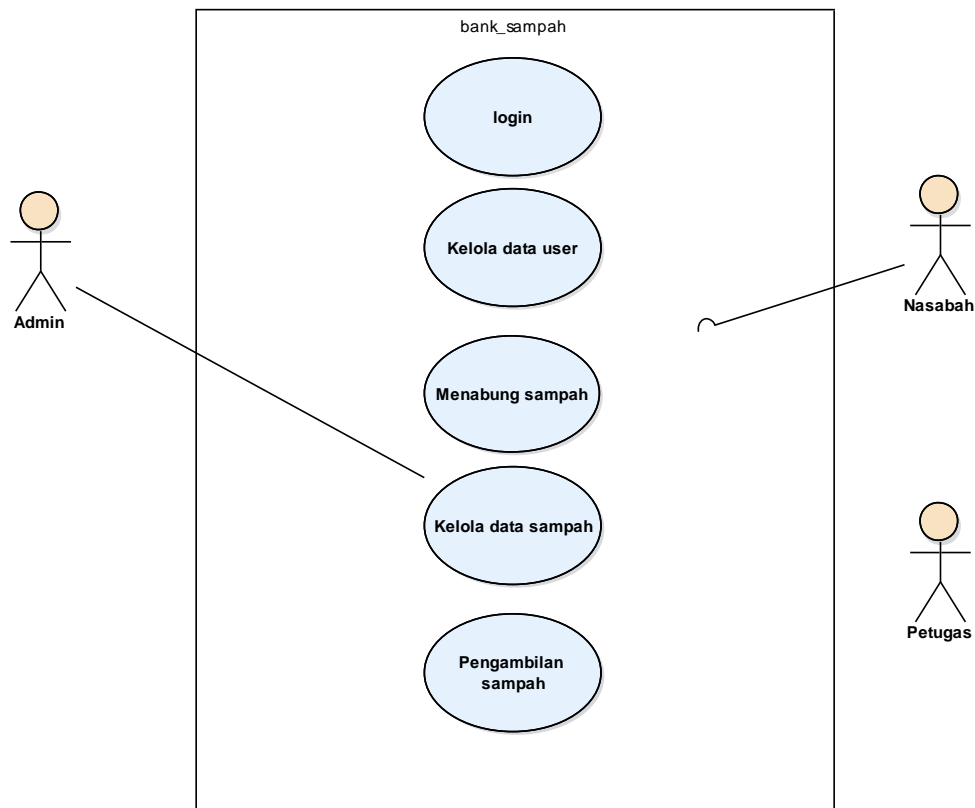
Aplikasi yang akan dibuat ini digunakan ketika ingin menabung sampah dan melakukan penjemputan sampah, adapun User yang dilibatkan antara lain : Nasabah (Masyarakat) dan petugas kebersihan.

8.2 Perancangan

8.2.1 Use Case Diagram

Use case diagram mendeskripsikan sebuah interaksi antara satu atau lebih actor dengan sistem informasi yang akan dibuat. Use case diagram digunakan untuk mengetahui

fungsi apa saja yang ada di dalam sebuah sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi itu.



Gambar 8.1 Use Case Diagram

8.2.1.1 Definisi Aktor

Pada bagian ini akan dideskripsikan actor-aktor yang terlibat dalam Aplikasi Bank Sampah

Tabel 8.1 Definisi Aktor

No	Aktor	Deskripsi
1.	Admin	<div>- Mengelola data sampah</div> <div>- Mengelola data user</div>

- Mengatur jadwal pengambilan sampah
- 2. Nasabah
 - Menabung sampah
- 3. Petugas
 - Kelola data sampah

8.2.1.2 Definisi Use Case

Use case digunakan untuk mengetahui fungsi apa aja yang ada didalam sebuah sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi itu.

Tabel 8.2 Definisi Use Case

No	Use Case	Deskripsi
1.	Login	Merupakan aktifitas validasi user yang bisa melakukan akses data kedalam sistem.
2.	Kelola data user	Merupakan aktifitas untuk mengelola data sampah,
3	Kelola data sampah	Merupakan aktifitas untuk mengelola data nasabah
4	Menabung sampah	Merupakan aktifitas untuk meminta jadwal pengambilan sampah pada admin
.5.	Pengambilan sampah	Merupakan aktifitas untuk melakukan pengambilan sampah

8.2.1.3 Skenario Use Case

Skenario Use Case mendeskripsikan urutan langkah-langkah dalam proses bisnis, baik yang dilakukan aktor terhadap sistem maupun yang dilakukan oleh sistem terhadap actor.

Tabel 8.3 Scenario Diagram Login

Identifikasi	
Nomor	1
Nama	Login
Tujuan	Merupakan aktifitas validasi yang bisa melakukan akses data ke dalam sistem.
Deskripsi	
Aktor	Admin, Nasabah dan Petugas
Skenario Utama	
Kondisi Awal	form sudah tersedia
Aksi Aktor	Reaksi Sistem
1. Masukan <i>username</i> dan <i>password</i> 2. Jika nasabah dan petugas belum memiliki akun maka harus melakukan registrasi akun terlebih dahulu	3. Mencocokkan data <i>login</i> dengan data Admin, nasabah dan petugas pada basis data 4. Menyimpan data yang telah melkaukan <i>register</i> pada basis data
Kondisi Akhir	Jika admin, nasabah dan petugas memasukan <i>username</i> dan <i>password</i> berhasil, maka akan masuk ke halamannya masing-masing

Tabel 8.4 Scenario Diagram Kelola Data User

Identifikasi	
Nomor	2
Nama	Kelola data user
Tujuan	Merupakan aktifitas untuk mengelola data nasabah
Deskripsi	
Aktor	Admin

Skenario Utama	
Kondisi Awal	form sudah tersedia
Aksi Aktor	Reaksi Sistem
1. Admin mengelola data role 2. Admin mengelola aktivasi nasabah	3. Sistem menjalankan perintah untuk mengelola data role 4. Sistem menjalankan perintah aktivasi
Kondisi Akhir	Jika admin telah melakukan aktivasi pada nasabah maka nasabah akan mendapatkan no rekening secara otomatis

Tabel 3.5 Scenario Diagram Kelola Sampah

Identifikasi	
Nomor	3
Nama	Kelola Data Sampah
Tujuan	Merupakan aktifitas untuk mengelola data sampah
Deskripsi	
Aktor	Admin
Skenario Utama	
Kondisi Awal	form sudah tersedia
Aksi Aktor	Reaksi Sistem
1. Admin memilih perintah (edit, tambah, dan delete)	2. Sistem menjalankan perintah yang dipilih oleh admin
Kondisi Akhir	Jika perintah berhasil dijalankan maka sistem akan menyimpan, mengubah dan menghapus datantergantung perintah apa yang admin jalankan .

Tabel 3.6 Scenario Diagram menabung Sampah

Identifikasi	
Nomor	4
Nama	Menabung Sampah
Tujuan	Merupakan aktifitas untuk meminta jadwal pengambilan sampah
Deskripsi	
Aktor	Nasabah
Skenario Utama	
Kondisi Awal	form sudah tersedia
Aksi Aktor	Reaksi Sistem
1. Nasabah mengatur tanggal pengambilan sampah	1. Sistem menjalankan perintah yang dipilih oleh nasabah
Kondisi Akhir	Jika perintah berhasil dijalankan maka tanggal yang telah ditetapkan oleh nasabah akan tersimpan pada baasis data dan data tersebut akan di tampilkan pada halaman pengambilan sampah .

Tabel 3.7 Scenario Diagram Pengambilan Sampah

Identifikasi	
Nomor	5
Nama	Pengambilan Sampah
Tujuan	Merupakan aktifitas untuk mengelola pengambilan sampah
Deskripsi	
Aktor	Admin dan petugas
Skenario Utama	
Kondisi Awal	form sudah tersedia

Aksi Aktor	Reaksi Sistem
1. Admin mengatur pembagian pengambilan sampah pada petugas 2. Petugas menginputkan data sampah	3. Sistem menjalankan perintah yang dipilih oleh Admin 4. Sistem akan menjalankan perintah yang dipilih oleh petugas
Kondisi Akhir	Jika perintah yang dipilih oleh admin maka sistem akan menampilkan data pengambilan sampah pada petugas dan jika perintah yang telah dijalankan oleh petugas maka sistem akan menyimpan data.

BAB IX

KODE PROGRAM

9.1 Controler

9.1.1 Admin.php

Kode pada admin.php ini dibuat untuk menampilkan form admin ketika admin telah selesai melakukan proses login dan tidak valid. Dalam dashboard admin ini terdapat form role, aktivasi nasabah, input sampah, jemput sampah, history penjemputan, penarikan saldo nasabah, dan info harga sampah. Admin akan mengelola seluruh form yang ada pada dashboard admin.

```
<?php
defined('BASEPATH') or exit('No direct script access allowed');

class Admin extends CI_Controller
{
    public function __construct()
    {
        parent::__construct();
        cek_role();
        $this->load-
>model('Aktivasi_model', 'aktivasi', true);
        $this->load->model('Admin_model', 'admin', true);
        $this->load->helper(array('form', 'url'));
    }

    public function index()
    {
        $data['title'] = 'Dashboard';
```

```

        $data['user'] = $this->db-
>get_where('user', ['email' => $this->session-
>userdata('email')])->row_array();

        $this->load->view('templates/header', $data);
        $this->load->view('templates/sidebar', $data);
        $this->load->view('templates/topbar', $data);
        $this->load->view('admin/index', $data);
        $this->load->view('templates/footer');
    }

    public function role()
    {
        $data['title'] = 'Role';
        $data['user'] = $this->db-
>get_where('user', ['email' => $this->session-
>userdata('email')])->row_array();

        $data['role'] = $this->db->get('user_role')-
>result_array();

        $this->form_validation-
>set_rules('role', 'Role', 'required');

        if ($this->form_validation->run() == false) {
            $this->load->view('templates/header', $data);
            $this->load-
>view('templates/sidebar', $data);
            $this->load->view('templates/topbar', $data);
            $this->load->view('admin/role', $data);
            $this->load->view('templates/footer');
        } else {
            $this->db-
>insert('user_role', ['role' => $this->input-
>post('role')]);
            $this->session-
>set_flashdata('message', '<div class="alert alert-
success mx-
auto" role="alert">Role baru telah ditambahkan!</div>');

```

```

        redirect('admin/role');
    }
}

public function roleAccess($role_id)
{
    $data['title'] = 'Role Access';
    $data['user'] = $this->db->get_where('user', ['email' => $this->session->userdata('email')])->row_array();

    $data['role'] = $this->db->get_where('user_role', ['id' => $role_id])->row_array();

    $this->db->where('id != 1');
    $data['menu'] = $this->db->get('user_menu')->result_array();

    $this->load->view('templates/header', $data);
    $this->load->view('templates/sidebar', $data);
    $this->load->view('templates/topbar', $data);
    $this->load->view('admin/role-access', $data);
    $this->load->view('templates/footer');
}

public function changeAccess()
{
    $menu_id = $this->input->post('menuId');
    $role_id = $this->input->post('roleId');

    $data = [
        'role_id' => $role_id,
        'menu_id' => $menu_id
    ];

    $result = $this->db->get_where('user_access_menu', $data);

    if ($result->num_rows() < 1) {

```

```

        $this->db->insert('user_access_menu', $data);
    } else {
        $this->db->delete('user_access_menu', $data);
    }

    $this->session->
>set_flashdata('message', '<div class="alert alert-
success mx-
auto" role="alert">Akses telah berubah!</div>');
    }

    public function hapus($id)
    {
        $this->load->model('Admin_model', 'role');
        $this->role->hapusRole($id);
        $this->session->
>set_flashdata('message', '<div class="alert alert-
success mx-
auto" role="alert">Role berhasil dihapus</div>');
        redirect('admin/role');
    }

    public function edit()
    {
        $this->form_validation->
>set_rules('role', 'Role', 'required');

        if ($this->form_validation->run() == false) {
            $this->session->
>set_flashdata('message', '<div class="alert alert-
danger mx-auto" role="alert">Role gagal di edit!</div>');
            redirect('admin/role');
        } else {
            $data = [
                'role' => $this->input->post('role')
            ];
            $this->db->where('id', $_POST['id']);
            $this->db->update('user_role', $data);
        }
    }
}

```

```

        $this->session->
>set_flashdata('message', '<div class="alert alert-
success mx-
auto" role="alert">Role berhasil di edit!</div>');
        redirect('admin/role');
    }
}

public function aktivasi()
{
    $data['title'] = 'Aktivasi Nasabah';
    $data['user'] = $this->db->
>get_where('user', ['email' => $this->session->
>userdata('email')])>row_array();

    $data['hasil'] = $this->db->
>get_where('user_detail', ['role_id' => '2'])>
>result_array();
    $data['aktivasi'] = $this->admin->getAktivasi();

    $this->load->view('templates/header', $data);
    $this->load->view('templates/sidebar', $data);
    $this->load->view('templates/topbar', $data);
    $this->load->view('admin/aktivasi', $data);
    $this->load->view('templates/footer');
}

public function editrek($id)
{
    $this->load->model('Aktivasi_model', 'aktivasi');
    $this->aktivasi->editRek($id);
    $this->session->
>set_flashdata('message', '<div class="alert alert-
success mx-auto" role="alert">Akun Diaktifkan!</div>');
    redirect('admin/aktivasi');
}

public function sampah()
{

```

```

        $data['title'] = 'Data Sampah';
        $data['user'] = $this->db-
>get_where('user', ['email' => $this->session-
>userdata('email')])->row_array();

        $data['sampah'] = $this->db->get('sampah')-
>result_array();

        $this->form_validation-
>set_rules('nama', 'Jenis Sampah', 'required', [
            'required' => 'Jenis Sampah harus diisi!'
        ]);

        $this->form_validation-
>set_rules('harga', 'Harga', 'required', [
            'required' => 'Harga harus diisi!'
        ]);

        if ($this->form_validation->run() == false) {
            $this->load->view('templates/header', $data);
            $this->load-
>view('templates/sidebar', $data);
            $this->load->view('templates/topbar', $data);
            $this->load->view('admin/sampah', $data);
            $this->load->view('templates/footer');
        } else {
            $query = [
                'nama' => $this->input->post('nama'),
                'harga' => $this->input->post('harga'),
            ];
            $this->db->insert('sampah', $query);
            $this->session-
>set_flashdata('message', '<div class="alert alert-
success mx-
auto" role="alert">Data Sampah telah ditambahkan!</div>');
            ;

            redirect('admin/sampah');
        }
    }
}

```

```

    public function hapusS($id)
    {
        $this->load->model('Admin_model', 'sampah');
        $this->sampah->hapusSampah($id);
        $this->session->
>set_flashdata('message', '<div class="alert alert-
success mx-
auto" role="alert">Data Sampah berhasil dihapus</div>');
        redirect('admin/sampah');
    }

    public function editS()
    {
        $this->form_validation->
>set_rules('nama', 'Jenis Sampah', 'required', [
            'required' => 'Jenis Sampah harus diisi!'
        ]);

        if ($this->form_validation->run() == false) {
            $this->session->
>set_flashdata('message', '<div class="alert alert-
danger mx-
auto" role="alert">Data Sampah gagal di edit!</div>');
            redirect('admin/sampah');
        } else {
            $data = [
                'nama' => $this->input->post('nama'),
                'harga' => $this->input->post('harga')
            ];
            $this->db->where('id', $_POST['id']);
            $this->db->update('sampah', $data);
            $this->session->
>set_flashdata('message', '<div class="alert alert-
success mx-
auto" role="alert">Data Sampah berhasil di edit!</div>');
            redirect('admin/sampah');
        }
    }
}

```

```

    public function jemput()
    {
        $data['title'] = 'Permintaan Jemput Sampah dari N
asabah';
        $data['user'] = $this->db-
>get_where('user', ['email' => $this->session-
>userdata('email')])->row_array();

        $data['jemput'] = $this->admin->getAll();
        $data['nama'] = $this->admin->getJadwal();
        $data['petugas'] = $this->db-
>get_where('user', ['role_id' => '3'])->result_array();

        $this->load->view('templates/header', $data);
        $this->load->view('templates/sidebar', $data);
        $this->load->view('templates/topbar', $data);
        $this->load->view('admin/jemput', $data);
        $this->load->view('templates/footer');
    }

    public function aturP()
    {
        $this->form_validation-
>set_rules('petugas', 'Petugas', 'required', [
            'required' => 'Petugas Harus dipilih!'
        ]);

        if ($this->form_validation->run() == false) {
            $this->session-
>set_flashdata('message', '<div class="alert alert-
danger mx-
auto" role="alert">Data Penjemputan gagal di jadwalkan!</
div>');
            redirect('admin/jemput');
        } else {
            $query = [
                'req_id' => $this->input->post('req_id'),

```



```

        'petugas_id' => $this->input-
>post('petugas')
    ];
    $this->db->insert('jadwal', $query);
    $this->session-
>set_flashdata('message', '<div class="alert alert-
success mx-
auto" role="alert">Data Penjemputan berhasil di jadwalkan
!</div>');
    redirect('admin/jemput');
}
}

public function histori()
{
    $data['title'] = 'Histori Penjemputan Sampah';
    $data['user'] = $this->db-
>get_where('user', ['email' => $this->session-
>userdata('email')])->row_array();

    $data['histori'] = $this->admin->getHistori();

    $this->load->view('templates/header', $data);
    $this->load->view('templates/sidebar', $data);
    $this->load->view('templates/topbar', $data);
    $this->load->view('admin/histori', $data);
    $this->load->view('templates/footer');
}

public function tarik()
{
    $data['title'] = 'Permintaan Penarikan Saldo';
    $data['user'] = $this->db-
>get_where('user', ['email' => $this->session-
>userdata('email')])->row_array();

    $data['penarikan'] = $this->admin->penarikan();

    $this->load->view('templates/header', $data);

```

```

        $this->load->view('templates/sidebar', $data);
        $this->load->view('templates/topbar', $data);
        $this->load->view('admin/tarik');
        $this->load->view('templates/footer');
    }

    public function saldoTarik()
    {
        $this->form_validation-
>set_rules('id', 'Id', 'required', [
            'required' => 'id Harus dipilih!'
        ]);

        if ($this->form_validation->run() == false) {
            $this->session-
>set_flashdata('message', '<div class="alert alert-
danger mx-
auto" role="alert">Penarikan Saldo gagal di Validasi!</di
v>');

            redirect('admin/tarik');
        } else {
            $query = [
                'user_id' => $this->input-
>post('user_id'),
                'penarikan' => $this->input-
>post('penarikan'),
                'tanggal' => time()
            ];
            $query2 = [
                'status' => 'disetujui'
            ];
            $this->db->insert('tabungan', $query);
            $this->db->where('id', $this->input-
>post('id'));
            $this->db->update('tarik', $query2);
            $this->session-
>set_flashdata('message', '<div class="alert alert-
success mx-
auto" role="alert">Penarikan Saldo berhasil!</div>');

```

```

        redirect('admin/tarik');
    }
}

public function hapusTarik()
{
    $id = $this->input->post('id');
    $this->admin->hapus_tarik($id);
    $this->session->
>set_flashdata('message', '<div class="alert alert-
info mx-
auto" role="alert">Data Penarikan tidak disetujui!</div>'
);
    redirect('admin/tarik');
}
}

```

9.1.2 Auth.php

Kode pada auth.php ini dibuat untuk menampilkan form login, register dan logout. Form ini akan muncul pada tampilan awal ketika user mengklik login pada halaman aplikasi bank sampah dan ketika user belum memiliki akun maka akan melakukan proses registrasi terlebih dahulu.

```

<?php
defined('BASEPATH') or exit('No direct script access allo
wed');

class Auth extends CI_Controller
{

    public function __construct()
    {

```

```

        parent::__construct();
        $this->load->library('form_validation');
    }

    public function index()
    {
        if ($this->session->userdata('email')) {
            redirect('user');
        }

        $this->form_validation->
>set_rules('email', 'Email', 'required|trim|valid_email',
    [
        'required' => 'email harus diisi!'
    ]);
        $this->form_validation->
>set_rules('password', 'Password', 'required|trim', [
        'required' => 'password harus diisi!'
    ]);

        if ($this->form_validation->run() == false) {
            $data['title'] = 'Login Page';
            $this->load->
>view('templates/auth_header', $data);
            $this->load->view('auth/login');
            $this->load->view('templates/auth_footer');
        } else {

            $this->_login();
        }
    }

    private function _login()
    {
        $email = $this->input->post('email');
        $password = $this->input->post('password');

        $user = $this->db->
>get_where('user', ['email' => $email])>row_array();

```

```

        if ($user) {

            if ($user['is_active'] == 1) {
                if (password_verify($password, $user['password'])) {

                    $data = [
                        'id' => $user['id'],
                        'email' => $user['email'],
                        'role_id' => $user['role_id']
                    ];
                    $this->session->set_userdata($data);
                    if ($user['role_id'] == 1) {
                        redirect('admin/aktivasi');
                    }
                    if ($user['role_id'] == 3) {
                        redirect('petugas');
                    } else {
                        redirect('tabungan');
                    }
                } else {
                    $this->session->
                    >set_flashdata('message', '<div class="alert alert-
                    danger mx-auto" role="alert">Password salah!</div>');
                    redirect('auth');
                }
            } else {
                $this->session->
                >set_flashdata('message', '<div class="alert alert-
                danger mx-auto" role="alert">Email tidak aktif</div>');
                redirect('auth');
            }
        } else {
            $this->session->
            >set_flashdata('message', '<div class="alert alert-
            danger mx-
            auto" role="alert">Email tidak terdaftar!</div>');
            redirect('auth');
        }
    }
}

```

```

    }

    public function registration()
    {
        $this->load->model('Auth_model', 'auth');

        if ($this->session->userdata('email')) {
            redirect('user');
        }

        $this->form_validation->
>set_rules('name', 'Name', 'required|trim', [
            'required' => 'nama harus diisi!',
        ]);
        $this->form_validation->
>set_rules('alamat', 'Alamat', 'required', [
            'required' => 'alamat harus diisi!',
        ]);
        $this->form_validation->
>set_rules('email', 'Email', 'required|trim|valid_email|i
s_unique[user.email]', [
            'required' => 'email harus diisi!',
            'valid_email' => 'format email salah!',
            'is_unique' => 'email ini sudah terdaftar!'
        ]);

        $this->form_validation->
>set_rules('password1', 'Password', 'required|trim|min_le
ngth[3]|matches[password2]', [
            'required' => 'password harus diisi!',
            'matches' => 'password tidak sama!',
            'min_length' => 'password terlalu pendek!'
        ]);
        $this->form_validation->
>set_rules('password2', 'Password', 'required|trim|matche
s[password1]', [
            'required' => 'password harus diisi!'
        ]);
    }

```

```

        if ($this->form_validation->run() == false) {
            $data['title'] = 'Pendaftaran Bank Sampah';
            $this->load-
>view('templates/auth_header', $data);
            $this->load->view('auth/registration');
            $this->load->view('templates/auth_footer');
        } else {
            $data = [
                'name' => htmlspecialchars($this->input-
>post('name', true)),
                'email' => htmlspecialchars($this->input-
>post('email', true)),
                'image' => 'default.jpg',
                'password' => password_hash($this->input-
>post('password1'), PASSWORD_DEFAULT),
                'role_id' => $this->input-
>post('radiob', true),
                'is_active' => 1,
                'date_created' => time(),
            ];
            $data1 = [
                'no_rek' => 0,
                'nama' => htmlspecialchars($this->input-
>post('name', true)),
                'alamat' => htmlspecialchars($this-
>input->post('alamat', true)),
                'scan_ktp' => 'default.png',
                'scan_kk' => 'default.png',
                'user_id' => $this->auth-
>create('user', $data),
                'role_id' => $this->input-
>post('radiob', true),
            ];
            $insert1 = $this->auth-
>create('user_detail', $data1);
            $this->session-
>set_flashdata('message', '<div class="alert alert-
success mx-

```

```

auto" role="alert">Selamat! Akun anda telah berhasil dibuat. Silahkan Login!</div>');
        redirect('auth');
    }
}

public function logout()
{
    $this->session->unset_userdata('email');
    $this->session->unset_userdata('role_id');

    $this->session->set_flashdata('message', '<div class="alert alert-success mx-auto" role="alert">Anda telah berhasil Logout</div>');
    redirect('auth');
}

public function blocked()
{
    $this->load->view('auth/blocked');
}
}

```

9.1.3 Data.php

Kode pada data.php ini dibuat untuk menampilkan form data detail dari akun nasabah yang telah melakukan proses registrasi. Pada form itu akun nasabah diharuskan memasukan data seperti foto KTP (Kartu Tanda Penduduk) dan KK (Kartu Keluarga). Maka data-data tersebut akan di periksa oleh admin setelah data itu sesuai maka admin akan melakukan proses aktivasi


```

<?php
defined('BASEPATH') or exit('No direct script access allowed');

class Data extends CI_Controller
{
    public function __construct()
    {
        parent::__construct();

        public function index()
        {
            $data['title'] = 'Data Detail';
            $data['user'] = $this->db->get_where('user', ['email' => $this->session->userdata('email')])->row_array();

            $data['detail'] = $this->db->get_where('user_detail', ['user_id' => $this->session->userdata('id')])->row_array();

            $this->form_validation->set_rules('nama', 'Nama Lengkap', 'required|trim', [
                'required' => 'nama harus diisi'
            ]);
            $this->form_validation->set_rules('alamat', 'Alamat', 'required|trim', [
                'required' => 'alamat harus diisi'
            ]);

            if ($this->form_validation->run() == false) {
                $this->load->view('templates/header', $data);
                $this->load->view('templates/sidebar', $data);
                $this->load->view('templates/topbar', $data);
                $this->load->view('data/index', $data);
                $this->load->view('templates/footer');
            } else {

```

```

        //untuk cek foto
        $upload_image1 = $_FILES['fotoktp']['name'];
        if ($upload_image1) {
            $config1['allowed_types'] = 'jpg|png|gif'
;
            $config1['max_size']      = '3000';
            $config1['upload_path']   = './assets/
img/scan/ktp/';

            $this->load->library('upload', $config1);
            $this->upload->initialize($config1);

            if ($this->upload-
>do_upload('fotoktp')) {
                $old_image1 = $data['user_detail']['
scan_ktp'];

                if ($old_image1 != 'default.png') {
                    unlink(FCPATH . ('assets/img/scan
/ktp/') . $old_image1);
                }

                $new_image1 = $this->upload-
>data('file_name');
                $this->db-
>set('scan_ktp', $new_image1);
            } else {
                echo $this->upload->display_errors();
            }
        }

        $upload_image2 = $_FILES['fotokk']['name'];
        if ($upload_image2) {
            $config2['allowed_types'] = 'jpg|png|gif'
;
            $config2['max_size']      = '3000';
            $config2['upload_path']   = './assets/
img/scan/kk/';

            $this->load->library('upload', $config2);

```

```

        $this->upload->initialize($config2);

        if ($this->upload->do_upload('fotokk')) {
            $old_image2 = $data['user_detail']['scan_kk'];

            if ($old_image2 != 'default.png') {
                unlink(FCPATH . ('assets/img/scan/kk/') . $old_image2);
            }

            $new_image2 = $this->upload->data('file_name');
            $this->db->set('scan_kk', $new_image2);
        } else {
            echo $this->upload->display_errors();
        }
    }
    $isi = [
        'nama' => $this->input->post('nama'),
        'alamat' => $this->input->post('alamat')
    ];
    $this->db->where('id', $_POST['id']);
    $this->db->update('user_detail', $isi);
    $this->db->set('name', $_POST['nama']);
    $this->db->where('email', $this->session->userdata('email'));
    $this->db->update('user');
    $this->session->set_flashdata('message', '<div class="alert alert-success mx-auto" role="alert">Profile anda telah diperbaharui!</div>');
    redirect('data/index');
}
}
}

```

9.1.4 Info.php

Kode pada inf.php ini dibuat untuk menampilkan form info harga sampah dimana admin, nasabah (masyarakat) dan petugas dapat melihat form info harga sampah

```
<?php
defined('BASEPATH') or exit('No direct script access allowed');

class Info extends CI_Controller
{
    public function __construct()
    {
        parent::__construct();
        cek_role();
    }

    public function index()
    {
        $data['title'] = 'Info - Harga Sampah';
        $data['user'] = $this->db->get_where('user', ['email' => $this->session->userdata('email')])->row_array();

        $data['info'] = $this->db->get('sampah')->result_array();

        $this->load->view('templates/header', $data);
        $this->load->view('templates/sidebar', $data);
        $this->load->view('templates/topbar', $data);
        $this->load->view('info/index', $data);
        $this->load->view('templates/footer');
    }
}
```

9.1.5 Menu.php

Kode pada menu.php ini dibuat untuk menampilkan form menu management. Form ini hanya dapat diakses oleh admin dan admin akan melakukan proses pengelolaan menu pada form menu tersebut.

```
<?php
defined('BASEPATH') or exit('No direct script access allowed');

class Menu extends CI_Controller
{
    public function __construct()
    {
        parent::__construct();
        cek_role();
    }
    public function index()
    {
        $data['title'] = 'Menu Management';
        $data['user'] = $this->db->get_where('user', ['email' => $this->session->userdata('email')])->row_array();

        $data['menu'] = $this->db->get('user_menu')->result_array();

        $this->form_validation->set_rules('menu', 'Menu', 'required', [
            'required' => 'Menu harus diisi'
        ]);

        if ($this->form_validation->run() == false) {
            $this->load->view('templates/header', $data);
            $this->load->view('templates/sidebar', $data);
```

```

        $this->load->view('templates/topbar', $data);
        $this->load->view('menu/index', $data);
        $this->load->view('templates/footer');
    } else {
        $this->db->
>insert('user_menu', ['menu' => $this->input->
>post('menu')]);
        $this->session->
>set_flashdata('message', '<div class="alert alert-
success mx-
auto" role="alert">Menu baru telah ditambahkan!</div>');
        redirect('menu');
    }
}

public function edit()
{
    $this->form_validation->
>set_rules('id', 'Id', 'required', [
        'required' => 'tidak ada id'
    ]);
    $this->form_validation->
>set_rules('menu', 'Menu', 'required');
    if ($this->form_validation->run() == FALSE) {
        $this->session->
>set_flashdata('message', '<div class="alert alert-
danger mx-auto" role="alert">Menu gagal di edit!</div>');
        redirect('menu');
    } else {
        $data = array(
            "menu" => $_POST['menu'],
        );
        $this->db->where('id', $_POST['id']);
        $this->db->update('user_menu', $data);
        $this->session->
>set_flashdata('message', '<div class="alert alert-
success mx-
auto" role="alert">Menu berhasil di edit!</div>');
        redirect('menu');
    }
}

```

```

    }
}

public function hapus($id)
{
    $this->load->model('Menu_model', 'menu');
    $this->menu->hapusMenu($id);
    $this->session->
    set_flashdata('message', '<div class="alert alert-
success mx-
auto" role="alert">Menu berhasil dihapus</div>');
    redirect('menu');
}

public function hapussm($id)
{
    $this->load->model('Menu_model', 'menu');
    $this->menu->hapusSubMenu($id);
    $this->session->
    set_flashdata('message', '<div class="alert alert-
success mx-
auto" role="alert">Submenu berhasil dihapus</div>');
    redirect('menu/submenu');
}

public function submenu()
{
    $data['title'] = 'Submenu Management';
    $data['user'] = $this->db->
    get_where('user', ['email' => $this->session->
    userdata('email')])->row_array();

    $this->load->model('Menu_model', 'menu');
    $data['submenu'] = $this->menu->getSubMenu();
    $data['menu'] = $this->db->get('user_menu')->
    result_array();

    $this->form_validation->
    set_rules('title', 'Title', 'required', [

```

```

        'required' => 'title harus diisi'
    ]);
    $this->form_validation-
>set_rules('menu_id', 'Menu', 'required', [
        'required' => 'menu harus diisi'
    ]);
    $this->form_validation-
>set_rules('url', 'Url', 'required', [
        'required' => 'url harus diisi'
    ]);
    $this->form_validation-
>set_rules('icon', 'Icon', 'required', [
        'required' => 'icon harus diisi'
    ]);

    if ($this->form_validation->run() == false) {

        $this->load->view('templates/header', $data);
        $this->load-
>view('templates/sidebar', $data);
        $this->load->view('templates/topbar', $data);
        $this->load->view('menu/submenu', $data);
        $this->load->view('templates/footer');
    } else {
        $data = [
            'title' => $this->input->post('title'),
            'menu_id' => $this->input-
>post('menu_id'),
            'url' => $this->input->post('url'),
            'icon' => $this->input->post('icon')
        ];
        $this->db->insert('user_sub_menu', $data);
        $this->session-
>set_flashdata('message', '<div class="alert alert-
success mx-
auto" role="alert">Submenu baru telah ditambahkan!</div>'
);
        redirect('menu/submenu');
    }
}

```



```

    }

    public function editism()
    {
        $this->form_validation->
>set_rules('id', 'Id', 'required', [
            'required' => 'tidak ada id'
        ]);
        $this->form_validation->
>set_rules('menu_id', 'Menu', 'required');
        $this->form_validation->
>set_rules('title', 'Title', 'required');
        $this->form_validation->
>set_rules('url', 'Title', 'required');
        $this->form_validation->
>set_rules('icon', 'Icon', 'required');

        if ($this->form_validation->run() == FALSE) {
            $this->session->
>set_flashdata('message', '<div class="alert alert-
danger mx-auto" role="alert">Menu gagal di edit!</div>');
            redirect('menu/submenu');
        } else {
            $data = [
                'title' => $this->input->post('title'),
                'menu_id' => $this->input->
>post('menu_id'),
                'url' => $this->input->post('url'),
                'icon' => $this->input->post('icon')
            ];
            $this->db->where('id', $_POST['id']);
            $this->db->update('user_sub_menu', $data);
            $this->session->
>set_flashdata('message', '<div class="alert alert-
success mx-
auto" role="alert">Menu berhasil di edit!</div>');
            redirect('menu/submenu');
        }
    }
}

```

```
}
```

9.1.6 Petugas.php

Kode pada petugas.php ini dibuat untuk menampilkan form petugas, ketika petugas berhasil melakukan proses login maka akan masuk ke form petugas tersebut.

```
<?php
defined('BASEPATH') or exit('No direct script access allowed');

class Petugas extends CI_Controller
{
    public function __construct()
    {
        parent::__construct();
        cek_role();
        $this->load->model('Petugas_model', 'petugas');
    }

    public function index()
    {
        $data['title'] = 'Jadwal Pengambilan';
        $data['user'] = $this->db->get_where('user', ['email' => $this->session->userdata('email')])->row_array();

        $data['ambil'] = $this->petugas->getAmbil();
        $data['jenis'] = $this->petugas->getJenis();
        $data['nasabah'] = $this->petugas->getNasabah();
        $data['petugas'] = $this->db->get('sampah')->result_array();

        $this->load->view('templates/header', $data);
```

```

        $this->load->view('templates/sidebar', $data);
        $this->load->view('templates/topbar', $data);
        $this->load->view('petugas/index', $data);
        $this->load->view('templates/footer');
    }

    public function input()
    {
        $data['title'] = 'Jadwal Pengambilan';
        $data['user'] = $this->db-
>get_where('user', ['email' => $this->session-
>userdata('email')])->row_array();

        $data['ambil'] = $this->petugas->getAmbil();
        $data['petugas'] = $this->db->get('sampah')-
>result_array();

        $setoranjmlh['v1'] = (int) $this->input-
>post('j_sampah', true);
        $setoranjmlh['v2'] = (int) $this->input-
>post('b_sampah', true);
        $setoran = $setoranjmlh['v1'] * $setoranjmlh['v2'
    ];

        $user_id = (int) $this->input->post('id', true);

        $this->form_validation-
>set_rules('j_sampah', 'Jenis Sampah', 'required', [
            'required' => 'Jenis Sampah harus diisi!'
        ]);
        $this->form_validation-
>set_rules('b_sampah', 'Berat Sampah', 'required', [
            'required' => 'Berat Sampah harus diisi!'
        ]);

        if ($this->form_validation->run() == false) {
            $this->session-
>set_flashdata('message', '<div class="alert alert-
danger mx-auto" role="alert">Pengambilan Gagal!!</div>');

```

```

        $this->load->view('templates/header', $data);
        $this->load-
>view('templates/sidebar', $data);
        $this->load->view('templates/topbar', $data);
        $this->load->view('petugas/index', $data);
        $this->load->view('templates/footer');
    } else {
        $query = [
            'user_id' => $user_id,
            'setoran' => $setoran,
            'tanggal' => time()
        ];
        $query2 = [
            'status' => 'diambil'
        ];
        $this->db->insert('tabungan', $query);
        $this->db->where('id', $_POST['req_id']);
        $this->db->update('req', $query2);
        $this->session-
>set_flashdata('message', '<div class="alert alert-
success mx-
auto" role="alert">Pengambilan berhasil!!</div>');
        redirect('petugas/index');
    }
}

public function histori()
{
    $data['title'] = 'Histori Pengambilan';
    $data['user'] = $this->db-
>get_where('user', ['email' => $this->session-
>userdata('email')])>row_array();

    $data['histori'] = $this->petugas->getHistori();

    $this->load->view('templates/header', $data);
    $this->load->view('templates/sidebar', $data);
    $this->load->view('templates/topbar', $data);
    $this->load->view('petugas/histori', $data);

```

```

        $this->load->view('templates/footer');
    }
}

```

9.1.7 Tabungan.php

Kode pada tabungan.php ini dibuat untuk menampilkan form saldo , pada form ini nasabah dapat melakukan pengecekan saldo dan melakukan penarikan saldo dikarenakan telah mempunyai no.rekening.

```

<?php
defined('BASEPATH') or exit('No direct script access allowed');

class Tabungan extends CI_Controller
{
    public function __construct()
    {
        parent::__construct();
        cek_role();
        $this->load->model('Tabungan_model', 'tabungan');
    }

    public function index()
    {
        $data['title'] = 'Tabungan';
        $data['user'] = $this->db->get_where('user', ['email' => $this->session->userdata('email')])->row_array();

        $data['tabungan'] = $this->tabungan->getTabungan();
        $data['nasabah'] = $this->tabungan->getNasabah();
        $data['total'] = $this->tabungan->getTotal();
    }
}

```

```

        $this->load->view('templates/header', $data);
        $this->load->view('templates/sidebar', $data);
        $this->load->view('templates/topbar', $data);
        $this->load->view('tabungan/index', $data);
        $this->load->view('templates/footer');
    }

    public function tarik()
    {
        $data['title'] = 'Meminta Penarikan Saldo';
        $data['user'] = $this->db-
>get_where('user', ['email' => $this->session-
>userdata('email')])->row_array();

        $data['nasabah'] = $this->db-
>get_where('user_detail', ['user_id' => $this->session-
>userdata('id')])->row_array();
        $data['histori'] = $this->tabungan->histori();

        $this->form_validation-
>set_rules('tarik', 'Jumlah Penarikan', 'required|trim',
[
            'required' => 'berapa jumlah yang ingin ditar
ik?'
        ]));

        if ($this->form_validation->run() == false) {
            $this->load->view('templates/header', $data);
            $this->load-
>view('templates/sidebar', $data);
            $this->load->view('templates/topbar', $data);
            $this->load->view('tabungan/tarik', $data);
            $this->load->view('templates/footer');
        } else {
            $isi = [
                'user_id' => $this->input-
>post('user_id'),

```

```

        'penarikan' => $this->input-
>post('tarik'),
        'tanggal' => time(),
        'status' => 'belum disetujui'
    ];
    $this->db->insert('tarik', $isi);
    $this->session-
>set_flashdata('message', '<div class="alert alert-
success mx-
auto" role="alert">Permintaan Penjemputan telah ditambahk
an!</div>');
    redirect('tabungan/tarik');
    }
}
}

```

9.1.8 User.php

Kode pada user.php ini dibuat untuk menampilkan form semua user baik itu admin, nasabah (masyarakat) dan petugas mereka semua dapat masuk ke form user tersebut. Dalam form user tersebut mereka dapat melihat profile, mengedit profile dan mengubah password sesuai apa yang user inginkan.

```

<?php
defined('BASEPATH') or exit('No direct script access allo
wed');

class User extends CI_Controller
{
    public function __construct()
    {
        parent::__construct();
    }
}

```

```

        cek_role();
    }
    public function index()
    {
        $data['title'] = 'My Profile';
        $data['user'] = $this->db->
>get_where('user', ['email' => $this->session->
>userdata('email')])->row_array();

        $this->load->view('templates/header', $data);
        $this->load->view('templates/sidebar', $data);
        $this->load->view('templates/topbar', $data);
        $this->load->view('user/index', $data);
        $this->load->view('templates/footer');
    }

    public function edit()
    {
        $data['title'] = 'Edit Profile';
        $data['user'] = $this->db->
>get_where('user', ['email' => $this->session->
>userdata('email')])->row_array();

        $this->form_validation->
>set_rules('name', 'Nama Lengkap', 'required|trim', [
            'required' => 'nama harus diisi'
        ]);

        if ($this->form_validation->run() == false) {
            $this->load->view('templates/header', $data);
            $this->load->
>view('templates/sidebar', $data);
            $this->load->view('templates/topbar', $data);
            $this->load->view('user/edit', $data);
            $this->load->view('templates/footer');
        } else {
            $name = $this->input->post('name');
            $email = $this->input->post('email');
            $nama = $this->input->post('name');

```



```

        $id = $this->input->post('id');

        //untuk cek foto
        $upload_image = $_FILES['foto']['name'];

        if ($upload_image) {
            $config['allowed_types'] = 'jpg|png|gif';
            $config['max_size']      = '3000';
            $config['upload_path']   = './assets/img/profile/';

            $this->load->library('upload', $config);

            if ($this->upload->do_upload('foto')) {
                $old_image = $data['user']['image'];
                if ($old_image != 'default.jpg') {
                    unlink(FCPATH . ('assets/img/profile/' . $old_image));
                }

                $new_image = $this->upload->data('file_name');
                $this->db->set('image', $new_image);
            } else {
                echo $this->upload->display_errors();
            }
        }

        $this->db->set('name', $name);
        $this->db->where('email', $email);
        $this->db->update('user');
        $this->db->set('nama', $nama);
        $this->db->where('user_id', $id);
        $this->db->update('user_detail');

        $this->session->set_flashdata('message', '<div class="alert alert-success mx-

```

```

auto" role="alert">Profile anda telah diperbaharui!</div>
');
        redirect('user');
    }
}

public function ubahPassword()
{
    $data['title'] = 'Ubah Password';
    $data['user'] = $this->db-
>get_where('user', ['email' => $this->session-
>userdata('email')])>row_array();

    $this->form_validation-
>set_rules('password_sekarang', 'Password Sekarang', 'req
uired|trim', [
        'required' => 'password sekarang harus diisi'
    ]);
    $this->form_validation-
>set_rules('password_baru1', 'Password Baru', 'required|t
rim|min_length[3]|matches[password_baru2]', [
        'required' => 'password baru harus diisi!',
        'min_length' => 'password terlalu pendek!'
    ]);
    $this->form_validation-
>set_rules('password_baru2', 'Konfirmasi Password Baru',
'required|trim|min_length[3]|matches[password_baru1]', [
        'required' => 'konfirmasi password harus diis
i!',
        'matches' => 'password tidak sama!'
    ]);

    if ($this->form_validation->run() == false) {
        $this->load->view('templates/header', $data);
        $this->load-
>view('templates/sidebar', $data);
        $this->load->view('templates/topbar', $data);
        $this->load-
>view('user/ubahpassword', $data);
    }
}

```

```

        $this->load->view('templates/footer');
    } else {
        $password_sekarang = $this->input-
>post('password_sekarang');
        $password_baru = $this->input-
>post('password_baru1');

        if (!password_verify($password_sekarang, $data[
'a[user]']['password'])) {
            $this->session-
>set_flashdata('message', '<div class="alert alert-
danger mx-
auto" role="alert">Masukan Password sekarang dengan benar
!</div>');
            redirect('user/ubahpassword');
        } else {
            if ($password_sekarang == $password_baru)
            {
                $this->session-
>set_flashdata('message', '<div class="alert alert-
danger mx-
auto" role="alert">Password tidak boleh sama!</div>');
                redirect('user/ubahpassword');
            } else {
                $password_hash = password_hash($password_baru, PASSWORD_DEFAULT);

                $this->db-
>set('password', $password_hash);
                $this->db->where('email', $this->session->userdata('email'));
                $this->db->update('user');

                $this->session-
>set_flashdata('message', '<div class="alert alert-
success mx-
auto" role="alert">Password telah diubah!</div>');
                redirect('user/ubahpassword');
            }
        }
    }
}

```

```

    }
}
}
}

```

9.1.9 Welcome.php

Kode pada welcome.php ini hanya sebagai contoh index pada tampilan halaman utama saja.

```

<?php
defined('BASEPATH') OR exit('No direct script access allowed');

class Welcome extends CI_Controller {

    /**
     * Index Page for this controller.
     *
     * Maps to the following URL
     *      http://example.com/index.php/welcome
     * - or -
     *      http://example.com/index.php/welcome/index
     * - or -
     * Since this controller is set as the default controller in
     * config/routes.php, it's displayed at http://example.com/
     *
     * So any other public methods not prefixed with an underscore will
     * map to /index.php/welcome/<method_name>
     * @see https://codeigniter.com/user_guide/general/urls.html
     */
    public function index()

```

```

    {
        $this->load->view('welcome_message');
    }
}

```

9.2 Models

9.2.1 Admin_model.php

Pada Admin_model.php ini memanggil beberapa fungsi method yaitu getAll(), getHistori(), getJadwal(), penarikan(), dan hapus_tarik(\$id)

```

<?php
defined('BASEPATH') or exit('No direct script access allowed');

class Admin_model extends CI_Model
{
    public function hapusRole($id)
    {
        $this->db->where('id', $id);
        $this->db->delete('user_role');
    }

    public function hapusSampah($id)
    {
        $this->db->where('id', $id);
        $this->db->delete('sampah');
    }

    public function getAktivasi()
    {
        $this->db->select('*');
        $this->db->from('user_detail');
    }
}

```

```

        $this->db->where('no_rek = "0" ');
        $query = $this->db-
>get_where('', ['role_id' => '2']);
        return $query->result_array();
    }

    public function getAll()
    {
        $this->db->select('req.*, user_detail.nama');
        $this->db->from('req');
        $this->db-
>where('req.status = "Belum diambil" ');
        $this->db-
>join('user_detail', 'req.user_id = user_detail.user_id')
;
        $query = $this->db->get();
        return $query->result();
    }

    public function getHistori()
    {
        $this->db-
>select('jadwal.*, req.*, user.name, user_detail.*');
        $this->db->from('jadwal', 'req');
        $this->db->where('req.status = "diambil" ');
        $this->db->join('req', 'jadwal.req_id = req.id');
        $this->db->join('user', 'req.user_id = user.id');
        $this->db-
>join('user_detail', 'req.user_id = user_detail.user_id')
;

        $query = $this->db->get();
        return $query->result();
    }

    public function getJadwal()
    {
        $this->db-
>select('req.*, user_detail.nama, jadwal.petugas_id');

```

```

        $this->db->from('req');
        $this->db-
>join('user_detail', 'req.user_id = user_detail.user_id')
;

        $this->db-
>join('jadwal', 'req.id = jadwal.req_id');
        $query = $this->db->get();
        return $query->result();
    }

    public function penarikan()
    {
        $this->db->select('tarik.*, user_detail.nama');
        $this->db->from('tarik');
        $this->db-
>join('user_detail', 'tarik.user_id = user_detail.user_id
');
        $this->db->order_by('tarik.id');
        $this->db-
>where('tarik.status = "belum disetujui"');
        $query = $this->db->get();
        return $query->result();
    }

    public function hapus_tarik($id)
    {
        $this->db->where('id', $id);
        $this->db->delete('tarik');
    }
}

```

9.2.2 Aktivasi_model.php

Pada Aktivasi_model.php ini memanggil beberapa fungsi method yaitu : getUser() dan editRek()

```

<?php
defined('BASEPATH') or exit('No direct script access allowed');

class Aktivasi_model extends CI_Model
{
    public function getUser()
    {
        # $this->db->order_by('user_detail.id', 'ASC');
        # return $this->db->from('user_detail')
        # -
        >join('user', 'user.id=user_detail.user_id')
        # ->get()
        # ->result();
        $this->db->select('*');
        $this->db->from('user_detail');
        $this->db->
        >join('user', 'user.id=user_detail.user_id');

        $query = $this->db->get();
        return $query->result();
    }
    function get_user($user_id)
    {
        $this->db->select('*');
        $this->db->from('user');
        $this->db->
        >join('user_detail', 'user_detail.user_id=user.id');
        $this->db->where('user.id', $user_id);
        $query = $this->db->get();
        return $query;
    }

    public function editRek($id)
    {
        $data = [
            'no_rek' => getAutoNumber('user_detail', 'no_rek', 'BS', 6),
        ];
    }
}

```



```

        $this->db->where('id', $id);
        $this->db->update('user_detail', $data);
    }
}

```

9.2.3 Auth_model.php

Pada auth_model.php memanggil beberapa fungsi method yaitu : create(\$table, \$data)

```

<?php
defined('BASEPATH') or exit('No direct script access allowed');

class Auth_model extends CI_Model
{
    public function create($table, $data)
    {
        $query = $this->db->insert($table, $data);
        return $this->db->insert_id();
    }
}

```

9.2.4 Menu_model.php

Pada ,Menu_model.php ini memanggil beberapa fungsi method yaitu : getSubMenu(), editMenu(\$id), hapusMenu(\$id) dan hapusSubMenu(\$id)

```

<?php
defined('BASEPATH') or exit('No direct script access allowed');

```

```

class Menu_model extends CI_Model
{
    public function getSubMenu()
    {
        $query = " SELECT `user_sub_menu`.*, `user_menu`.
`menu`
                FROM `user_sub_menu` JOIN `user_menu`
                ON `user_sub_menu`.`menu_id` = `user_m
enu`.`id`
                ";

        return $this->db->query($query)->result_array();
    }

    public function editMenu($id)
    {
        $isi = [
            'menu' => $this->input->post('editmenu')
        ];
        $this->db->where('id', $id);
        $this->db->update('user_menu', $isi);
    }

    public function hapusMenu($id)
    {
        $this->db->where('id', $id);
        $this->db->delete('user_menu');
    }

    public function hapusSubMenu($id)
    {
        $this->db->where('id', $id);
        $this->db->delete('user_sub_menu');
    }
}

```

9.2.5 Petugas_model.php

Pada petugas_model.php ini memanggil beberapa fungsi method yaitu : getJenis(), getAmbil(), getNasabah() dan getHistori()

```
<?php
defined('BASEPATH') or exit('No direct script access allowed');

class Petugas_model extends CI_Model
{
    public function getJenis()
    {
        $query = " SELECT *
                    FROM sampah";

        return $this->db->query($query)->result_array();
    }

    public function getAmbil()
    {
        $this->db->select('user_detail.*, req.tgl_jemput, req.id as req_id,
req.status, jadwal.petugas_id, user.id');
        $this->db->from('user_detail');
        $this->db->where('req.status != "diambil"');
        $this->db->join('req', 'user_detail.user_id = req.user_id');
        $this->db->join('jadwal', 'req.id = jadwal.req_id');
        $this->db->join('user', 'user_detail.user_id = user.id');
        $query = $this->db->get_where('', ['petugas_id' => $this->session->userdata('id')]);
        return $query->result();
    }
}
```

```

    }

    public function getNasabah()
    {
        $this->db->select('user_detail.*, tabungan.*');
        $this->db->from('user_detail');
        $this->db->
>join('tabungan', 'user_detail.id = tabungan.user_id');
        $query = $this->db->get();
        return $query->result();
    }

    public function getHistori()
    {
        $this->db->
>select('jadwal.*, req.*, user.name, user_detail.no_rek')
;
        $this->db->from('jadwal', 'req');
        $this->db->where('req.status = "diambil" ');
        $this->db->join('req', 'jadwal.req_id = req.id');
        $this->db->join('user', 'req.user_id = user.id');
        $this->db->
>join('user_detail', 'req.user_id = user_detail.user_id')
;

        $query = $this->db->
>get_where('', ['petugas_id' => $this->session-
>userdata('id')]);
        return $query->result();
    }
}

```

9.2.6 Rek_model.php

Pada rek_model.php ini memanggil beberapa fungsi method yaitu : buat_rek()

```

<?php
defined('BASEPATH') or exit('No direct script access allowed');

class Rek_model extends CI_Model
{
    function __construct()
    {
        parent::__construct();
    }

    public function buat_rek()
    {
        $this->db->select('RIGHT(user_detail.no_rek,6) as rek', false);
        $this->db->order_by('id', 'DESC');
        $this->db->limit(1);

        $query = $this->db->get('user_detail');

        if ($query->num_rows() != 0) {
            $data = $query->row();
            $rek = intval($data->rek) + 1;
        } else {
            $rek = 1;
        }
        $rek_max = str_pad($rek, 6, "0", STR_PAD_LEFT);
        $rek_jadi = "BS" . $rek_max;
        return $rek_jadi;
    }
}

```

9.2.7 Request_model.php

Pada Request_model.php ini memanggil beberapa fungsi method yaitu : historis()

```

<?php
defined('BASEPATH') or exit('No direct script access allowed');

class Request_model extends CI_Model
{
    public function histori()
    {
        $this->db->select('*');
        $this->db->from('req');
        $this->db->group_by('id');

        return $this->db->get_where('', ['user_id' => $this->session->userdata('id')])->result_array();
    }
}

```

9.2.8 Tabungan_model.php

Pada Tabungan_model.php ini memanggil beberapa fungsi method yaitu : getTabungan(), getTotal(), getNasabah() dan histori().

```

<?php
defined('BASEPATH') or exit('No direct script access allowed');

class Tabungan_model extends CI_Model
{
    public function getTabungan()
    {
        $this->db->select('tabungan.*, user.*');
        $this->db->from('tabungan');
    }
}

```

```

        $this->db-
>join('user', 'tabungan.user_id = user.id');
        return $this->db-
>get_where('', ['user_id' => $this->session-
>userdata('id')])->result_array();
    }

    public function getTotal()
    {
        $this->db-
>select('SUM(setoran) - SUM(penarikan) as total');
        $this->db->from('tabungan');
        return $this->db-
>get_where('', ['user_id' => $this->session-
>userdata('id')])->row()->total;
    }

    public function getNasabah()
    {
        $query = " SELECT `user_detail`.*, `user`.`id`
                    FROM `user_detail` JOIN `user`
                    ON `user_detail`.`user_id` = `user`.`i
d`
                    ";
        $result = $this->db-
>get_where('user_detail', ['user_id' => $this->session-
>userdata('id')]);
        return $result->row_array();
    }

    public function histori()
    {
        $this->db->select('*');
        $this->db->from('tarik');
        $this->db->group_by('id');

        return $this->db-
>get_where('', ['user_id' => $this->session-
>userdata('id')])->result_array();
    }

```

```
}  
}
```