

BUKU TUTORIAL SISTEM INFORMASI APPROVAL

“BUKU TUTORIAL SISTEM INFORMASI APPROVAL BERBASIS

WEB MENGGUNAKAN *FRAMEWORK CODEIGNITER* DENGAN

NOTIFIKASI E-MAIL”

(Studi Kasus: PT. X)

Buku ini dibuat untuk memenuhi persyaratan kelulusan
matakuliah Program Internship I



Dibuat Oleh,
1.16.4.085 Rahmi Roza

PROGRAM DIPLOMA IV TEKNIK INFORMATIKA
POLITEKNIK POS INDONESIA
BANDUNG
2019

Puji syukur saya panjatkan kepada Tuhan Yang Maha Esa, karena atas berkat dan rahmat-Nya, penulis dapat menyelesaikan laporan Internship I dengan judul “Sistem Informasi Planning Alat Produksi Berbasis Web (Simpro) Di PT. Telekomunikasi Indonesia (Telkom) (Sub Modul: Kaubis Dan Optima)”. Dalam penulisan, penulis menyadari bahwa masih banyak kekurangan dalam laporan ini mengingat keterbatasan pengetahuan penulis. Penulis megharapkan kritik dan saran yang sifatnya membangun dari pembaca. Untuk itu penulis mengucapkan terima kasih kepada:

1. M. Yusril Helmi Setyawan, S.Kom., M.Kom. selaku Ketua Program Studi D4 Teknik Informatika.
2. Nisa Hanum Harani, S.Si.,M.T. selaku Koordinator Internship 1.
3. Mohamad Nurkamal Fauzan, S.T.,M.T. selaku Pembimbing Internship 1.
4. M. Yusril Helmi Setyawan, S.Kom., M.Kom. sebagai dosen wali penulis yang sangat membantu dan memberikan dukungan menyelesaikan laporan Internship1.
5. Seluruh dosen program studi Teknik Informatika yang telah memberikan ilmu pengetahuan dan wawasan yang berguna bagi penulis.
6. Orang tua khususnya Ibu serta Kakak Penulis yang telah memberikan dukungan baik moril maupun materil.
7. Bapak Tatang Wiguna selaku pembimbing eksternal di PT. Telkom Lembong Bandung
8. Seluruh pengurus perpustakaan, yang telah menyediakan banyak referensi yang tentunya sangat mendukung penulis dalam penyelesaian laporan Internship 1 ini.

9. Teman – teman seperjuangan penulis Teknik Informatika 4C banyak membantu dalam usaha memperoleh data yang saya perlukan.
10. Seluruh pihak yang telah membantu dalam penyelesaian penyusunan laporan ini sesuai yang diharapkan yang tidak bisa penulis sebutkan satu persatu.

Akhir kata, penulis berharap Tuhan Yang Maha Esa berkenan membalas segala kebaikan semua pihak yang telah membantu. Semoga buku ini membawa manfaat bagi pengembangan ilmu.

Bandung, 19 Januari 2020

Penulis

DAFTAR ISI

DAFTAR ISI	4
DAFTAR GAMBAR	6
DAFTAR TABEL	7
BAB I	8
PENGENALAN SISTEM, INFORMASI, <i>PLANNING, web, approval</i>	8
1.1 Pengantar Sistem	8
1.2 Pengantar Informasi	17
1. Menjadi Sumber Pengetahuan Baru	23
2. Menghapus Ketidakpastian	24
3. Sebagai Media Hiburan	24
4. Sebagai Sumber Berita	24
5. Untuk Sosialisasi Kebijakan	24
6. Untuk Mempengaruhi Khalayak	24
7. Menyatukan Pendapat	25
1.3 Pengantar <i>Planning</i>	25
1.4 Pengantar <i>Web</i>	35
1.4.1 Sejarah <i>Web</i>	36
1.4.2 <i>HTTP (Hypertext Transfer Protocol)</i> dan <i>HTTPS (Hyper Text Transfer Protocol Secure)</i>	37
2. <i>HTTPS (Hyper Text Transfer Protocol Secure)</i>	39
1.4.3 <i>HTML (Hyper Text Markup Language)</i>	40
1.4.4 Hubungan <i>HTML</i> dan <i>PHP</i>	48
1.4.5 Pemahaman Pemrograman <i>Web</i>	49
1.4.6 Membuat Aplikasi Berbasis <i>Web</i>	50
1.4.7 Keunggulan Dan Kekurangan Aplikasi Berbasis <i>Web</i>	50
1.4.8 <i>Web Statis</i> dan <i>Web Dinamis</i>	51
1.4.9 Basis Data	52
1.5 Pengantar <i>Approval</i>	57
BAB II	58

2.1	<i>Framework</i>	58
2.2	<i>Framework CodeIgniter</i>	63
2.3	<i>E-Mail</i>	80
2.3.1	Apa itu <i>E-Mail</i> ?	80
2.3.2	Elemen-Elemen <i>E-Mail</i>	80
2.3.3	Perbandingan <i>E-Mail</i> Dengan Surat Biasa	81
2.3.4	Format <i>E-mail</i>	82
2.3.5	Tahapan Proses Pengiriman <i>E-mail</i>	83
2.3.6	Layanan <i>E-mail</i>	85
2.3.7	Komponen <i>E-mail</i>	86
2.3.8	<i>Multipurpose Internet Mail Extension</i>	89
2.3.9	Protokol <i>E-mail</i>	92
1.	<i>Simple Mail Transport Protocol</i>	92
2.4	Penjelasan <i>Tools</i> Yang Digunakan	96
2.4.1	<i>Visual Studio Code</i>	96
1.	Pengenalan <i>Visual Studio Code</i>	96
2.	Fitur-Fitur <i>Visual Studio Code</i>	97
3.	<i>Plugin Visual Studio Code</i>	98
2.4.2	<i>XAMPP</i>	100
2.4.3	<i>MySQL</i>	106
BAB III	111	
BAB IV	127	
BAB V	146	
BAB VI	211	
BAB VII	215	

DAFTAR GAMBAR

DAFTAR TABEL

BAB I

PENGENALAN SISTEM, INFORMASI, PLANNING, WEB, APPROVAL

Sistem, informasi, *planning*, *web* adalah beberapa istilah yang digunakan dalam penyusunan buku ini. Pada bab ini penulis akan memaparkan penjelasan mengenai ketiga istilah tersebut.

1.1 Pengantar Sistem

Secara umum, sistem adalah suatu kumpulan objek, unsur-unsur atau bagian yang memiliki arti berbeda-beda yang saling berhubungan, bekerja sama dan mempengaruhi satu sama lain serta memiliki keterikatan pada rencana atau *plane* yang sama dalam mencapai tujuan tertentu pada lingkungan yang kompleks.

Secara terminologi, sistem dipakai dalam berbagai macam cara yang luas sehingga sangat sulit untuk mendefinisikan atau mengartikannya sebagai sesuatu pernyataan yang merangkum seluruh penggunaannya dan cukup ringkas untuk memenuhi apa yang menjadi maksudnya.

1.1.1 Unsur-Unsur Sistem

Unsur-unsur sistem itu meliputi diantaranya, yaitu:

1. Obyek

Di dalam sistem terdapat sekumpulan objek, baik itu bersifat fisik ataupun abstrak dalam bentuk bagian, elemen, ataupun variable.

2. Atribut

Atribut merupakan sesuatu yang menentukan mutu maupun sifat kepemilikan suatu sistem serta objeknya.

3. Hubungan Internal

Hubungan internal yaitu setiap elemen yang saling terikat dalam satu kesatuan.

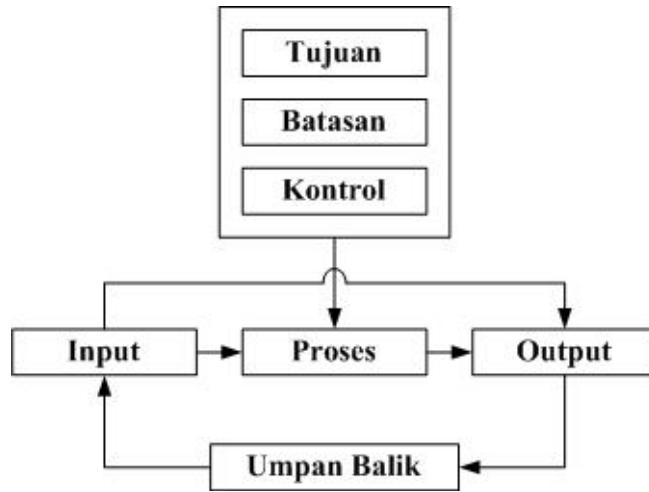
4. Lingkungan

Lingkungan berarti tempat ataupun wilayah di mana sistem itu berada.

1.1.2 Elemen-Elemen Pembentuk Sistem

Elemen pembentuk sistem itu meliputi diantaranya, yaitu:

1. Masukan, yang berarti semua yang masuk dalam sistem akan diproses, apakah itu merupakan objek fisik ataupun objek abstrak.
2. Tujuan, yang berarti sistem dibuat demi mencapai tujuan atau output tertentu yang ingin dicapai.
3. Proses, yaitu transformasi dari masukan yang diubah menjadi keluaran yang memiliki nilai lebih tinggi.
4. Keluaran atau *output*, yang merupakan hasil dari pemrosesan yang mana wujudnya dapat berbentuk informasi, cetakan laporan, sara, produk, dan lain sebagainya.
5. Batas, yakni sesuatu yang memisahkan antara sistem dan daerah yang berada di luar sistem. Batas inilah yang akan menentukan ruang lingkup, konfigurasi, dan banyak hal lain sebagainya.
6. Pengendalian dan umpan balik, yang mana mekanismenya bisa dilakukan dengan menggunakan *feedback* terhadap keluaran untuk mengendalikan elemen masukan atau proses.
7. Lingkungan, yaitu segala sesuatu di luar sistem yang memiliki pengaruh terhadap sistem, baik yang menguntungkan ataupun merugikan.



Gambar 1.1 Elemen-Elemen Sistem

1.1.3 Jenis-Jenis Sistem

1. Berdasarkan Keterbukaan

Jenis sistem yang pertama adalah berdasarkan keterbukaan, maka sistem itu terbagi menjadi dua yakni sistem terbuka dan sistem tertutup:

- i. Sistem Terbuka, yakni suatu sistem yang bisa dipengaruhi oleh pihak luar karena adanya akses yang terbuka.
- ii. Sistem Tertutup, yakni kebalikan dari sistem terbuka yaitu sistem yang tidak dapat dipengaruhi oleh pihak luar dikarenakan akses yang tertutup.

2. Berdasarkan Komponen

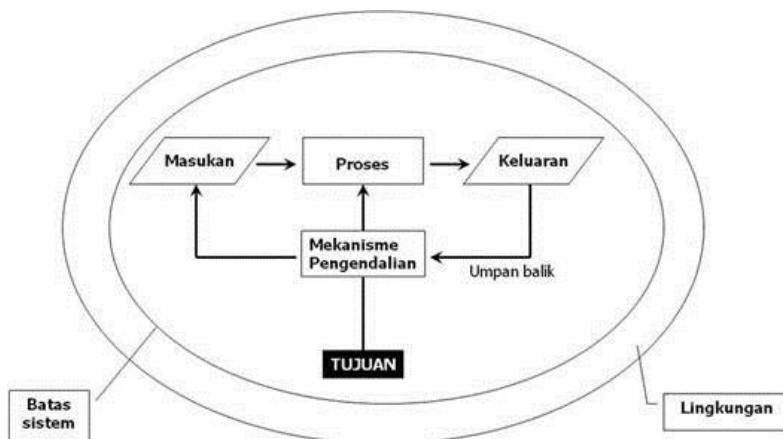
Berdasarkan komponennya sistem dibagi atas dua jenis. Yakni sistem fisik dan non-fisik. Perbedaan keduanya adalah sebagai berikut:

- i. Sistem Fisik, yakni suatu sistem yang mempunyai komponen energi dan materi.

- ii. Sistem non-fisik, yakni suatu sistem yang berbentuk abstrak, contohnya: dapat berupa ide, konsep, dan gagasan serta banyak contoh lainnya.

1.1.4 Karakter Sistem

Karakter sistem adalah sistem yang mempunyai komponen-komponen, batas sistem, penghubung, masukan, keluaran, pengolahan dan sasaran.



Gambar 1.2 Karakteristik Sistem

1. Komponen

Elemen-elemen yang lebih kecil yang disebut *sub sistem*, misalkan sistem computer terdiri dari sub sistem perangkat keras, perangkat lunak dan manusia.

Elemen-elemen yang lebih besar yang disebut *supra sistem*, misalkan apabila perangkat keras adalah sebuah sistem yang memiliki sub sistem *CPU*, perangkat *I/O* dan memori, maka supra sistem perangkat keras adalah sistem computer.

2. *Boundary* (Batasan Sistem)

Batas sistem merupakan daerah yang membatasi suatu sistem dengan sistem yang lainnya atau dengan lingkungan luarnya. Batas sistem ini menunjukkan ruang lingkup dari sistem tersebut.

3. *Environment* (Lingkungan Luar Sistem)

Lingkungan dari sistem adalah apapun di luar batas dari sistem yang mempengaruhi operasi sistem. Lingkungan luar sistem dapat bersifat menguntungkan dan dapat juga bersifat merugikan untuk sistem itu sendiri. Lingkungan luar yang menguntungkan merupakan energi dari sistem dari sistem dan demikian harus tetap dijaga dan dipelihara. Sedang lingkungan luar yang harus merugikan harus ditahan dan dikendalikan, jika tidak akan menganggu kelangsungan hidup dari sistem itu sendiri.

4. *Interface* (Penghubung Sistem)

Penghubung merupakan media perantara antar sub sistem. Melalui penghubung ini memungkinkan sumber-sumber daya mengalir dari satu subsistem ke subsistem lainnya. *Output* dari satu sub sistem akan menjadi input untuk subsistem yang lainnya dengan melalui penghubung. Dengan penghubung satu subsistem dapat berinteraksi dengan sub sistem yang lainnya membentuk satu kesatuan.

5. *Input* (Masukan)

Masukan adalah energi yang dimasukkan ke dalam sistem. Masukan dapat berupa *maintenance input* dan *sinyal input*. *Maintenance input* adalah energi yang dimasukkan supaya sistem tersebut dapat beroperasi. *Sinyal input* adalah energi yang diproses untuk didapatkan keluaran.

6. *Output* (Keluaran)

Keluaran adalah hasil dari energi yang diolah dan diklasifikasikan menjadi keluaran yang berguna dan sisa pembuangan. Keluaran dapat merupakan masukan untuk subsistem yang lain atau kepada supra sistem

7. Proses (Pengolahan Sistem)

Suatu sistem dapat mempunyai suatu bagian pengolah atau sistem itu sendiri sebagai pengolahnya. Pengolah yang akan merubah masukan menjadi keluaran. Suatu sistem produksi akan mengolah masukan berupa bahan baku dan bahan-bahan yang lain menjadi keluaran berupa barang jadi.

8. *Objective and Goal* (Sasaran dan Tujuan Sistem)

Suatu sistem pasti mempunyai tujuan atau sasaran. Kalau suatu sistem tidak mempunyai sasaran, maka operasi sistem tidak akan ada gunanya. Sasaran dari sistem sangat menentukan sekali masukan yang dibutuhkan sistem dan keluaran yang akan dihasilkan sistem. Suatu sistem dikatakan berhasil bila mengenai sasaran atau tujuannya.

1.1.5 Klasifikasi Sistem

Klasifikasi sistem adalah suatu bentuk kesatuan antara satu komponen dengan komponen lainnya, karena tujuan dari sebuah sistem memiliki akhir tujuan yang berbeda untuk setiap kasus ataupun perkara yang terjadi di dalam setiap sistem tersebut. Sehingga sistem dapat diklasifikasikan sebagai berikut:

1. Sistem Abstrak (*abstract system*), adalah sistem yang berupa pemikiran atau ide-ide yang tidak tampak secara fisik. Sedangkan sistem fisik (*physical system*) merupakan sistem yang ada secara fisik.

2. Sistem Alamiah (*natural system*), adalah sistem yang terjadi melalui proses alam, yang mana tidak dibuat oleh manusia. Sedangkan sistem buatan manusia (*human made system*) merupakan sistem yang melibatkan interaksi antara manusia dengan mesin.
3. Sistem Tertentu (*deterministic system*), adalah sistem yang beroperasi dengan tingkah laku yang sudah dapat diprediksi. Sedangkan sistem tak tentu (*probabilistic system*) adalah sistem yang kondisi masa depannya tidak bisa diprediksi karena mengandung unsur probabilitas.
4. Sistem Tertutup (*closed system*), merupakan sistem yang tidak berhubungan dan tidak terpengaruh dengan lingkungan luar. Sedangkan sistem terbuka (*open system*), merupakan sistem yang tidak berhubungan dan terpengaruh dengan lingkungan luar.

1.1.6 Pelaku Sistem

Pelaku sistem terdiri dari 7 kelompok, diantaranya sebagai berikut:

1. Pemakai, pada umumnya 3 ada jenis pemakai, yaitu operasional, pengawas dan eksekutif.
2. Manajemen, pada umumnya terdiri dari 3 jenis manajemen, yaitu manajemen pemakai yang bertugas menangani pemakaian dimana sistem baru diterapkan, manajemen sistem yang terlibat dalam pengembangan sistem itu sendiri dan manajemen umum yang terlibat dalam strategi perencanaan sistem dan sistem pendukung pengambilan keputusan. Kelompok manajemen biasanya terlibat dengan keputusan yang berhubungan dengan orang, waktu dan uang.
3. Pemeriksa, ukuran dan kerumitan sistem yang dikerjakan dan bentuk alami organisasi dimana sistem tersebut diimplementasikan dapat menentukan kesimpulan perlu tidaknya pemeriksa. Pemeriksa

biasanya menentukan segala sesuatunya berdasarkan ukuran-ukuran standar yang dikembangkan pada banyak perusahaan sejenis.

4. Penganalisa sistem, fungsi-fungsinya antara lain yaitu:
 - a. Arkeolog yaitu yang menelusuri bagaimana sebenarnya sistem lama berjalan, bagaimana sistem tersebut dijalankan dan segala hal yang menyangkut sistem lama.
 - b. Inovator, yaitu yang membantu mengembangkan dan membuka wawasan pemakai bagi kemungkinan-kemungkinan lain.
 - c. Mediator, yaitu yang menjalankan fungsi komunikasi dari semua level, antara lain: pemakai, manajer, programmer, pemeriksa dan pelaku sistem yang lainnya yang mungkin belum punya sikap dan cara pandang yang sama.
 - d. Pimpinan proyek, penganalisa sistem haruslah personil yang lebih berpengalaman dari programmer atau desainer. Selain itu mengingat penganalisa sistem umumnya ditetapkan terlebih dahulu dalam suatu pekerjaan sebelum yang lain bekerja, adalah hal yang wajar jika penanggung jawab pekerjaan menjadi porsi penganalisa sistem.
5. Pendesain sistem merupakan sistem menerima hasil penganalisa sistem berupa kebutuhan pemakai yang tidak berorientasi pada teknologi tertentu, yang kemudian ditransformasikan ke desain arsitektur tingkat tinggi dan dapat diformulasikan oleh programmer.
6. Programmer mengerjakan dalam bentuk program dari hasil desain yang telah diterima dari pendesain.
7. Personel pengoperasian bertugas dan bertanggungjawab di pusat komputer misalnya jaringan, keamanan perangkat keras, keamanan perangkat lunak, pencetakan dan *backup*. Pelaku ini mungkin tidak

diperlukan bila sistem yang berjalan tidak besar dan tidak membutuhkan klasifikasi khusus untuk menjalankan sistem.

1.1.7 Sistem Berdasarkan Prinsip

Sistem berdasarkan prinsip dasar secara umum terbagi dalam beberapa macam, diantaranya yaitu:

1. Sistem terspesialisasi adalah sistem yang sulit diterapkan pada lingkungan yang berbeda, misalnya sistem biologi, ikan yang dipindahkan ke darat.
2. Sistem besar adalah sistem yang sebagian besar sumber dayanya berfungsi melakukan perawatan harian, misalnya dinosaurus sebagai sistem biologi menghabiskan sebagian besar masa hidupnya dengan makan dan makan.
3. Sistem sebagai bagian dari sistem lain adalah sistem selalu merupakan bagian dari sistem yang lebih besar, dan dapat terbagi menjadi sistem yang lebih kecil.
4. Sistem berkembang merupakan sistem yang walaupun tidak berlaku bagi semua sistem tetapi hampir semua sistem selalu berkembang.

1.1.8 Analisis Sistem

Penganalisa sistem merupakan bagian dari tim yang berfungsi mengembangkan sistem yang memiliki daya guna tinggi dan memenuhi kebutuhan pemakai akhir. Pengembangan ini dipengaruhi sejumlah hal, yaitu:

1. Produktifitas, saat ini dibutuhkan sistem yang lebih banyak, lebih bagus dan lebih cepat. Hal ini membutuhkan lebih banyak programmer dalam penganalisa sistem yang berkualitas, kondisi kerja ekstra, kemampuan pemakai untuk mengambangkan sendiri, bahasa

- pemrograman yang lebih baik, perawatan sistem yang lebih baik (umumnya 50 % sampai 70 % sumber daya digunakan untuk perawatan sistem), disiplin teknis pemakaian perangkat lunak dan perangkat pengembangan sistem yang terotomasi.
2. Realibilitas, waktu yang dihabiskan untuk testing sistem secara umum menghabiskan 50% dari waktu total pengembangan sistem. Dalam kurun waktu 30 tahun sejumlah sistem yang digunakan di berbagai perusahaan mengalami kesalahan dan ironisnya sangat tidak mudah untuk mengubahnya. Jika terjadi kesalahan, ada dua cara yang bisa dilakukan, yaitu melakukan pelacakan sumber kesalahan dan harus menemukan cara untuk mengoreksi kesalahan tersebut dengan mengganti program, menghilangkan sejumlah statement lama atau menambahkan sejumlah statement baru.
 3. Maintabilitas, perawatan mencakup:
 - a. Modifikasi sistem sesuai perkembangan perangkat keras untuk meningkatkan kecepatan pemrosesan (yang memegang peranan penting dalam pengoperasian sistem)
 - b. Modifikasi sistem sesuai perkembangan kebutuhan pemakai. Antara 50% sampai 80% pekerjaan yang dilakukan pada kebanyakan pengembangan sistem dilakukan untuk revisi, modifikasi, konversi, peningkatan dan pelacakan kesalahan.

1.2 Pengantar Informasi

Informasi adalah data yang diolah menjadi bentuk yang sangat berguna untuk membuat sebuah keputusan. Informasi berguna untuk membuat keputusan karena informasi dapat menurunkan ketidakpastian pada data, karena berdasarkan informasi itu para pengelola dapat mengetahui kondisi

obyektif. Informasi tersebut merupakan hasil dari pengolahan data atau fakta yang sudah dikumpulkan menggunakan metode tertentu.



Gambar 1.3 Ilustrasi Informasi

1.2.1 Ciri-Ciri Informasi

Ciri-ciri informasi yang berkualitas bisa dilihat sebagai berikut:

1. Akurat, artinya informasi mencerminkan keadaan sebenarnya.
2. Tepat waktu, artinya informasi harus ada saat diperlukan.
3. Relevan, informasi yang diberikan harus sesuai dengan yang dibutuhkan.
4. Lengkap, artinya informasi harus utuh, tidak setengah-setengah.

1.2.2 Jenis-Jenis Informasi

1. ***Absolute Information***, merupakan “pohonnya” sebuah informasi yaitu jenis informasi yang disajikan dengan sebuah jaminan serta tidak membutuhkan penjelasan lebih lanjut.
2. ***Substitutional Information***, merupakan jenis yang merujuk pada kasus dimana konsep informasi digunakan untuk sejumlah informasi.
3. ***Philosophic Information***, merupakan jenis informasi yang berkaitan dengan perasaan dan infomasi manusia. Aadnya infomasi ini bergantung kepada orang yang menyajikannya.

4. ***Objective Information***, merupakan jenis informasi yang merujuk pada karakter logis informasi tertentu.
5. ***Cultural Information***, merupakan informasi yang memberikan tekanan pada dimensi *cultural* (budaya).

1.2.3 Contoh Informasi

1. Kenaikan harga BBM adalah Rp. 500 per liter untuk jenis pertamax.
2. Minuman jus wortel kaya akan vitamin A.
3. Harga laptop dan *smartphone* terbaru dengan harga di atas 2 juta.
4. Harga satu kilogram telur adalah 22.000 per kilonya.

1.2.4 Sumber Data atau Informasi

Sumber infomasi bisa didapat dari:

1. Lingkungan kerja
2. Lembaga pendidikan
3. Media masa
4. Instansi pemerintah
5. Masyarakat

1.2.5 Cara Mendapatkan Data atau Informasi

Ada beberapa cara untuk mendapatkan data atau informasi,diantaranya sebagai berikut:

1. Hasil penelitian yang dilakukan sebelumnya.
2. Data yang telah lewat dengan memperhatikan *trend* dan taksiran di masa depan.
3. Mengambil dari pusat data seperti Badan Pusat Statistik (BPS) dan Pusat Data Informasi Pertanian (Pustadin).
4. Media elektronik seperti televisi, radi, dan internet.

5. Media cetak seperti buku, majalah, karya ilmiah, koran, proposal, dan lain sebagainya.
6. Forum, seperti seminar, pelatihan, dan pendidikan.

1.2.6 Kualitas Informasi

Kualitas informasi ditentukan oleh berapa faktor yaitu sebagai berikut:

1. Keakuratan dan teruji kebenarannya

Informasi harus bebas dari kesalahan-kesalahan serta tidak menyesatkan.
2. Kesempurnaan Informasi

Informasi harus disajikan dengan lengkap tanpa pengurangan dan penambahan, serta adanya pengubahan.
3. Tepat Waktu

Informasi harus disajikan tepat waktu, karena menjadi dasar dalam pengambilan sebuah keputusan nantinya.
4. Relevansi

Informasi akan mempunyai manfaat dengan nilai yang sangat tinggi, jika informasi tersebut dapat diterima oleh mereka yang membutuhkan.
5. Mudah dan Cerah

Apabila cara serta biaya untuk memperoleh informasi sulit serta mahala, maka orang menjadi tidak berminat untuk memperolehnya, atau akan mencari alternatif lainnya untuk memperoleh infomasi itu sendiri.
6. Kualitas suatu informasi tergantung dari 3 hal, yaitu:
 - i. Akurat, berarti informasi harus bebas dari kesalahan-kesalahan dan harus jelas mencerminkan maksudnya.

- ii. Tepat pada waktunya, berarti informasi yang diterima tidak boleh terlambat atau dengan kata lain harus tepat waktu.
- iii. Relevan, berarti informasi tersebut mempunai manfaat dari pemakainya.

1.2.7 Tujuan Klasifikasi Data atau Informasi

- 1. Melindungi perjanjian kontrak (infromasi) dengan mitra bisnis atau konsumennya.
- 2. Memberikan pengamanan yang sesuai, sehingga menghemat sumber daya organisasi dan membuat pengelolaan infromasi menjadi efisien dan efektif.
- 3. Membantu meningkatkan kualitas data/infromasi yang digunakan sebagai bahan untuk mengambil keputusan.

1.2.8 Manfaat Data atau Informasi

Data atau Informasi memiliki manfaat yaitu:

- 1. Meningkatkan wawasan dan pengetahuan organisasi.
- 2. Mengurangi resiko kesalahan dalam pengambilan keputusan.
- 3. Menggambarkan kondisi yang terjadi di masa kini.
- 4. Memberi gambaran *trend* atau kecendrungan di masa depan.
- 5. Mengurangi ketidak pastian kondisi karena adanya keimangsiuran fenomena.
- 6. Menjadi dasar bagi pemecahan masalah.
- 7. Menghasilkan arus kerja menjadi lebih efektif dan efisien.
- 8. Meningkatkan citra positif perusahaan.
- 9. Menambah relasi.
- 10. Meningkatkan kepercayaan pemegang saham.
- 11. Memberi arahan bagi promosi yang lebih jelas.

12. Menjadi dasar pertanggung jawaban atas segala tindakan yang sudah diambil.
13. Memberikan bukti, bukan kesan, isu, atau opini dari pihak lain.

1.2.9 Komponen Informasi

Komponen-komponen informasi meliputi:

1. ***Root of Information***, yaitu komponen inti datu informasi berada pada tahap keluaran pertama sebuah proses pengolahan data yang biasanya disampaikan oleh orang pertama.
2. ***Bar of Information***, yaitu merupakan badan/batangnya dari informasi yang disajikan dan memerlukan informasi pendukung, agar informasi inti dapat diketahui secara utuh. Contoh : headline surat kabar agar pembaca jelas maka harus membaca informasi selanjutnya.
3. ***Branch of Information***, yaitu informasi dapat dipahami apabila informasi sebelumnya telah dipahami. Misalnya, ketika kita membaca glosarium atau indeks ketika membaca sebuah buku.
4. ***Stick of Information***, yaitu komponen informasi yang sederhana dari cabang informasi. Bentuk dari informasi ini biasanya berbentuk pengayaan pengetahuan, kedudukannya hanya sebagai pelengkap, terhadap informasi yang ada.
5. ***Bud of Information***, yaitu komponen informasi yang sifatnya semi mikro namun sangat dibutuhkan, sehingga diwaktu mendatang informasi ini akan berkembang dan dicari orang, misalnya informasi tentang multiple intelligence, hypoteaching, kurikulum masa depan, pembelajaran abad ke 21, dan lain-lain.
6. ***Leaf of Information***. yaitu merupakan informasi pelindung untuk menjelaskan kondisi dan situasi ketika informasi itu mucul ke

permukaan, seperti informasi tentang prakiraan cuaca, prakiraan kemarau panjang, prakiraan gempa atau gerhana matahari/bulan.

1.2.10 Perubahan Data Menjadi Informasi

Pemrosesan data (Inggris: *data processing*) adalah jenis pemrosesan yang dapat mengubah data menjadi informasi atau pengetahuan. Pemrosesan data ini sering menggunakan komputer sehingga bisa berjalan secara otomatis. Setelah diolah, data ini biasanya mempunyai nilai yang informatif jika dinyatakan dan dikemas secara terorganisir dan rapi, maka istilah pemrosesan data sering dikatakan sebagai sistem informasi.

Kedua istilah ini mempunyai arti yang hampir sama, pemrosesan data mengolah dan memanipulasi data mentah menjadi informasi (hasil pengolahan), sedangkan sistem informasi memakai data sebagai bahan masukan dan menghasilkan informasi sebagai produk keluaran. Pada saat ini kegiatan Data Processing sudah semakin luas, baik yang berorientasi kepada ilmu pengetahuan, komersil/bisnis maupun kegiatan pemerintahan, sehingga data yang diolahpun akan bermacam-macam sesuai dengan bidang pekerjaan tersebut.

1.2.11 Fungsi Informasi

1. Menjadi Sumber Pengetahuan Baru

Informasi valid yang didapatkan oleh seseorang dapat menjadi pengetahuan baru dan menambah wawasan di bidang tertentu. Misalnya informasi mengenai cara mengatasi masalah kesehatan yang didapatkan dari konten di internet.

Mungkin informasi tersebut adalah sesuatu yang umum dan sudah banyak diketahui orang. Namun, mungkin saja ada seseorang yang belum mengetahui informasi tersebut.

2. Menghapus Ketidakpastian

Kurangnya informasi tentang sesuatu akan menimbulkan ketidakpastian. Untuk menghapus ketidak pastian tersebut maka diperlukan informasi lengkap dan valid dari sumber terpercaya.

3. Sebagai Media Hiburan

Informasi juga dapat berfungsi sebagai media hiburan bagi masyarakat. Misalnya informasi mengenai objek wisata di suatu tempat yang disajikan dengan bahasa dan gambar-gambar yang menarik.

4. Sebagai Sumber Berita

Suatu informasi mengenai hal tertentu bisa dipakai sebagai sumber berita yang disampaikan kepada khalayak. Misalnya, informasi tentang Asian Games yang didapatkan dari media Televisi, Radio, dan situs berita online.

5. Untuk Sosialisasi Kebijakan

Informasi adalah komponen penting dalam berkomunikasi dengan pihak lain. Salah satunya adalah untuk menyampaikan suatu kebijakan dari pemerintah kepada masyarakat yang dilakukan dengan cara sosialisasi.

6. Untuk Mempengaruhi Khalayak

Penyampaian informasi melalui media massa biasanya dilakukan untuk mempengaruhi khalayak. Misalnya informasi mengenai suatu produk melalui Televisi yang tujuannya agar masyarakat mengenal dan tertarik untuk menggunakannya.

7. Menyatukan Pendapat

Di era media sosial seperti sekarang ini, sangat mudah untuk menyampaikan pendapat ke ruang publik. Namun, tidak semua pendapat tersebut sesuai dengan fakta yang ada.

Adanya informasi yang valid dari sumber terpercaya akan bermanfaat untuk menilai setiap pendapat yang dikemukakan di ruang publik apakah sesuai dengan informasi tersebut.

1.3 Pengantar *Planning*

Planning atau perencanaan adalah fungsi dasar (fundamental) manajemen, karena *organizing*, *directing*, dan *controlling* pun harus terlebih dahulu direncanakan. Hasil perencanaan baru akan diketahui di masa depan. Agar resiko yang ditanggung itu relatif kecil, hendaknya semua kegiatan, tindakan, dan kebijakan direncanakan terlebih dahulu. Perencanaan ini adalah masalah “memilih”, artinya memilih tujuan, dan cara terbaik untuk mencapai tujuan tersebut dari beberapa alternatif yang ada. Tanpa alternatif, perencanaan pun tidak ada. Dengan kata lain perencanaan merupakan kumpulan dari beberapa keputusan.

Perencanaan diproses oleh perencanaan (*planner*), hasilnya menjadi rencana (*plan*). Perencanaan dan rencana sangat penting karena:

1. Tanpa perencanaan dan rencana berarti tidak ada tujuan yang ingin dicapai.
2. Tanpa perencanaan dan rencana tidak ada pedoman pelaksanaan sehingga banyak pemborosan.
3. Rencana adalah dasar pengendalian, karena tanpa ada rencana pengendalian tidak dapat dilakukan.
4. Tanpa perencanaan dan rencana berarti tidak ada keputusan dan proses manajemen pun tidak ada.



Gambar 1.4 Ilustrasi Planning

1.3.1 Maksud Perencanaan (*Purpose of Planning*)

1. Perencanaan adalah salah satu fungsi manajer yang meliputi seleksi atas alternatif-alternatif tujuan, kebijaksanaan-kebijaksanaan, prosedur-prosedur dan program-program.
2. Perencanaan pada asasnya adalah memilih dan persoalan perencanaan timbul, jika suatu alternatif cara bertindak ditemukan.
3. Perencanaan, sebagai besar merupakan usaha membuat hal-hal terjadi sebagaimana yang dikehendaki.
4. Perencanaan adalah suatu proses pemikiran, penentuan tindakan-tindakan secara sadar berdasarkan keputusan-keputusan menyangkut tujuan, fakta, dan ramalan.
5. Perencanaan adalah usaha menghindari kekosongan tugas, tumpang tindih, dan meningkatkan efektifitas potensi yang dimiliki.

1.3.2 Tujuan Perencanaan (*Objective of Planning*)

1. Perencanaan bertujuan untuk menentukan tujuan, kebijakan-kebijakan, prosedur, dan program serta memberikan pedoman cara-cara pelaksanaan yang efektif dalam mencapai tujuan.
2. Perencanaan bertujuan untuk menjadikan tindakan ekonomis, karena semua potensi yang dimiliki terarah dengan baik pada tujuan.
3. Perencanaan adalah satu usaha untuk memperkecil risiko yang dihadapi pada masa yang akan datang.
4. Perencanaan menyebabkan kegiatan-kegiatan dilakukan secara teratur dan bertujuan.
5. Perencanaan memberikan gambaran yang jelas dan lengkap tentang seluruh pekerjaan
6. Perencanaan membantu penggunaan suatu alat pengukuran hasil kerja.
7. Perencanaan menjadi suatu landasan untuk pengendalian.
8. Perencanaan merupakan usaha untuk menghindari *mismanagement* dalam penempatan karyawan.
9. Perencanaan membantu peningkatan daya guna dan hasil guna organisasi.

1.3.3 Asas-Asas Perencanaan (*Principle of Planning*)

1. *Principle of contribution to objective*

Setiap perencanaan dan segala perubahannya harus ditujukan kepada pencapaian tujuan

2. *Principle of efficiency of planning*

Suatu perencanaan efisien, jika perencanaan itu dalam pelaksanaannya dapat mencapai tujuan dengan biaya uang sekecil-kecilnya.

3. *Principle of primacy of planning (asas pengutamaan perencanaan)*

Perencanaan adalah keperluan utama para pemimpin dan fungsi-fungsi lainnya, *organizing, staffing, directing, dan controlling*.

4. *Principle of pervasiveness of planning (asas pemerataan perencanaan)*

Asas pemerataan perencanaan memegang peranan penting mengingat pemimpin pada tingkat tinggi banyak mengerjakan perencanaan dan bertanggung jawab atas berhasilnya rencana itu.

5. *Principle of planning premise (asas patokan perencanaan)*

Patokan-patokan perencanaan sangat berguna bagi ramalan, sebab premis-premis perencanaan dapat menunjukkan kejadian-kejadian yang akan datang.

6. *Principle of policy frame work (asas kebijaksanaan pola kerja)*

Kebijaksanaan ini mewujudkan pola kerja, prosedur-prosedur kerja, dan program-program kerja tersusun.

7. *Principle of timing (asas time)*

Adalah perencanaan waktu yang relative singkat dan tepat

8. *Principle of planning Communcation*

Perencanaan dapat disusun dan dikoordinasikan dengan baik, jika setiap orang bertanggung jawab terhadap pekerjaannya dan memperoleh penjelasan yang memadai mengenai bidang yang akan dilaksanakannya.

9. *Principle of alternative (asas alternative)*

Alternative ada pada setiap rangkaian kerja dan perencanaan meliputi pemilihan rangkaian alternative dalam pelaksanaan pekerjaan, sehingga tercapai tujuan yang telah ditetapkan.

10. *Principle of limiting factor* (asas pembatasan faktor)

Dalam pemilihan alternative-alternative, pertama harus ditujukan pada faktor-faktor yang strategis dan dapat membantu pemecahan masalah. Asas alternative dan pembatasan faktor merupakan syarat mutlak dalam penetapan keputusan.

11. *The commitment principle* (asas keterikatan)

Perencanaan harus memperhitungkan jangka waktu keterikatan yang diperlukan untuk pelaksanaan pekerjaan.

12. *The principle of flexibility* (asas fleksibilitas)

Perencanaan yang efektif memerlukan fleksibilitas, tetapi tidak berarti mengubah tujuan.

13. *The principle of navigation change* (asas ketetapan arah)

Perencanaan yang efektif memerlukan pengamatan yang terus-menerus terhadap kejadian-kejadian yang timbul dalam pelaksanaannya untuk mempertahankan tujuan.

14. *Principle of strategic planning* (asas perencanaan strategis)

Dalam kondisi tertentu manajer harus memilih tindakan-tindakan yang diperlukan untuk menjamin pelaksanaan rencana agar tujuan tercapai.

1.3.4 Macam-Macam Perencanaan

Macam-macam perencanaan dalam pengantar manajemen dibagi menjadi 3 yaitu:

1. Perencanaan Organisasi

- a. Perencanaan Strategis

Rencana strategis yaitu rencana yang dikembangkan untuk mencapai tujuan strategis. Tepatnya, rencana strategis adalah rencana umum yang mendasari keputusan alokasi sumber daya, prioritas, dan langkah-langkah tindakan yang diperlukan untuk mencapai tujuan strategis.

b. Perencanaan Taktis

Adalah rencana ditujukan untuk mencapai tujuan taktis, dikembangkan untuk mengimplementasikan bagian tertentu dari rencana strategis. Rencana strategis pada umumnya melibatkan manajemen tingkat atas dan menengah dan jika dibandingkan dengan rencana strategis, memiliki jangka waktu yang lebih singkat dan suatu fokus yang lebih spesifik dan nyata.

c. Perencanaan Operasional

Adalah rencana yang menitikberatkan pada perencanaan rencana taktis untuk mencapai tujuan operasional. Dikembangkan oleh manajer tingkat menengah dan tingkat bawah, rencana operasional memiliki fokus jangka pendek dan lingkup yang relatif lebih sempit. Masingmasing rencana operasional berkenaan dengan suatu rangkaian kecil aktivitas. Kami menjelaskan perencanaan dengan lebih mendekati pada bagian selanjutnya.

2. Perencanaan Operasional

- a. Rencana sekali pakai : dikembangkan untuk melaksanakan serangkaian tindakan yang mungkin tidak berulang di masa mendatang.
- b. Program : rencana sekali pakai untuk serangkaian aktivitas yang besar.
- c. Proyek : rencana sekali pakai untuk lingkup yang lebih sempit dan lebih tidak kompleks dibandingkan dengan program

- d. Perencanaan tetap : dikembangkan untuk aktivitas yang berulang secara teratur selama suatu periode waktu tertentu.
- e. Kebijakan : rencana tetap yang merinci respons umum organisasi terhadap suatu masalah atau situasi tertentu.
- f. Prosedur operasi standar : rencana tetap yang menguraikan langkah-langkah yang harus diikuti dalam situasi tertentu
- g. Aturan dan peraturan : rencana tetap yang mendeskripsikan dengan tepat bagaimana aktivitas tertentu dilaksanakan

3. Perencanaan Kontinjensi

Jenis perencanaan lain yang juga penting adalah perencanaan kontinjensi (*contingency planning*) yaitu penentuan serangkaian tindakan alternatif jika suatu rencana tindakan secara tidak terduga terganggu atau dianggap tidak sesuai lagi.

1.3.5 Hambatan Dalam Penetapan Perencanaan

1. Tujuan Yang Tidak Tepat

Tujuan yang tidak tepat mempunyai banyak bentuk. Membayar deviden yang besar kepada pemegang saham mungkin tidak jika dananya didapatkan dengan mengorbankan penelitian dan pengembangan tujuan mungkin juga tidak tepat jika tujuan tersebut tidak dapat dicapai. Jika Kmart menetapkan tujuan untuk memperoleh lebih banyak pendapatan dibanding Wal-Mart tahun depan, karyawan perusahaan mungkin. Tujuan juga tidak tepat jika tujuan itu menepatkan terlalu banyak penekanan pada ukuran kuantitatif maupun kalitatif dari keberhasilan.

2. Sistem Penghargaan Yang Tidak Tepat

Dalam beberapa lingkungan, sistem penghargaan yang tidak tepat merupakan hambatan dalam penetapan tujuan dan perencanaan

3. Lingkungan Yang Dinamis dan Kompleks

Sifat dari suatu lingkungan organisasi juga merupakan hambatan bagi penetapan tujuan dan perencanaan yang efektif. Perubahan yang cepat, inovasi teknologi, dan persaingan yang ketat juga dapat meningkatkan kesulitan bagi suatu organisasi untuk secara akurat mengukur kesempatan dan ancaman di masa mendatang

4. Kengganan Untuk Menetapkan Tujuan

Hambatan lain terhadap perencanaan yang efektif adalah tujuan bagi mereka sendiri dan untuk unit-unit yang merupakan tanggung jawab mereka. Alasan untuk ini mungkin adalah kurangnya rasa percaya diri atau takut akan kegagalan. Jika seorang manajer menetapkan suatu tujuan spesifik, ringkas, dan berhubungan dengan waktu, maka apakah ia mencapai atau tidak mencapai tujuan tersebut akan tampak nyata. Manajer yang secara sadar atau tidak sadar berusaha untuk menghindari tingkat tanggung jawab ini lebih mungkin untuk menghindari usaha perencanaan organisasi. Pfizer, suatu perusahaan farmasi besar, mengalami masalah karena manajernya tidak menetapkan tujuan untuk penelitian dan pengembangan. Sebagai akibatnya, organisasi tersebut jauh tertinggal di belakang karena manajer tidak memiliki cara untuk mengetahui seberapa efektif usaha penelitian dan pengembangan mereka sebenarnya.

5. Penolakan Terhadap Perubahan

Hambatan lain dalam menetapkan tujuan dan perencanaan adalah penolakan terhadap perubahan. Perencanaan pada intinya terkait dengan perubahan sesuatu dalam organisasi. Avon Products hampir membuat dirinya sendiri bangkrut beberapa tahun yang lalu karena perusahaan bersikeras melanjutkan kebijakan

pembayaran deviden yang besar kepada para pemegang sahamnya. Ketika laba mulai turun, manajer menolak memotong deviden dan mulai melakukan pinjaman untuk membayar deviden tersebut. Hutang perusahaan meningkat dari \$3 juta menjadi \$1, 1 miliar dalam waktu delapan tahun. Pada akhirnya, manajer terpaksa menyelesaikan masalah dan memotong deviden.

6. Keterbatasan

Keterbatasan (constraints) yang membatasi apa yang dapat dilakukan organisasi merupakan hambatan utama yang lain.

1.3.6 Mengatasi Hambatan Perencanaan

1. Pemahaman Maksud dan Tujuan

Salah satu cara terbaik untuk memperlancar penetapan tujuan dan proses perencanaan adalah dengan maksud dasarnya. Manajer seharusnya juga mengetahui bahwa terdapat keterbatasan pada efektivitas penetapan tujuan dan pembuatan rencana. Penetapan tujuan dan perencanaan yang efektif tidak selalu memastikan keberhasilan, penyesuaian dan pengecualian diharapkan dari waktu ke waktu.

2. Komunikasi dan Partisipasi

Meskipun mungkin dibuat pada tingkat tinggi, tujuan dan rencana tersebut harus dikomunikasikan kepada pihak yang lain dalam organisasi. Setiap orang yang terlibat dalam proses perencanaan seharusnya tahu landasan apa yang mendasari strategi fungsional, dan bagaimana strategi tersebut diintegrasikan dan dikoordinasikan. Orang-orang yang bertanggung jawab untuk mencapai tujuan dan mengimplementasikan rencana harus

didengar pendapatnya dalam mengembangkan strategi tersebut. Setiap orang hampir selalu memiliki informasi yang berharga untuk disumbangkan / dan karena mereka yang akan mengimplementasikan rencana / keterlibatan mereka sangat penting orang biasanya lebih berkomitmen pada rencana yang pembentukannya mereka bantu .bahkan ketika suatu organisasi agar bersifat sentralistik atau menggunakan staf perencanaan, manajer dari berbagai tingkat dalam organisasi seharusnya dilibatkan dalam proses perencanaan.

3. Konsistensi/Revisi/dan Pembaruan

Tujuan seharusnya konsisten baik secara horizontal maupun secara vertikal. Konsistensi horizontal berarti bahwa tujuan seharusnya konsisten diseluruh organisasi/ dari satu departemen ke departemen lainnya.Konsistensi vertikal berarti bahwa tujuan seharusnya konsisten dari atas hingga ke bawah organisasi : tujuan strategis, taktis, dan operasional harus selaras. Karena penetapan tujuan dan perencanaan merupakan proses yang dinamis, tujuan dan perencanaan juga harus direvisi dan diperbarui secara berkala. Banyak organisasi melihat perlunya merevisi dan memperbarui dengan frekuensi yang semakin sering.

4. Sistem Penghargaan Yang Efektif

Secara umum, orang seharusnya diberi penghargaan baik karena menetapkan tujuan dan rencana yang efektif, maupun karena berhasil mencapainya. Karena kegagalan terkadang berasal dari faktor-faktor di luar pengendalian manajemen, orang seharusnya dipastikan bahwa kegagalan dalam mencapai tujuan tidak akan selalu memiliki konsekuensi hukuman.

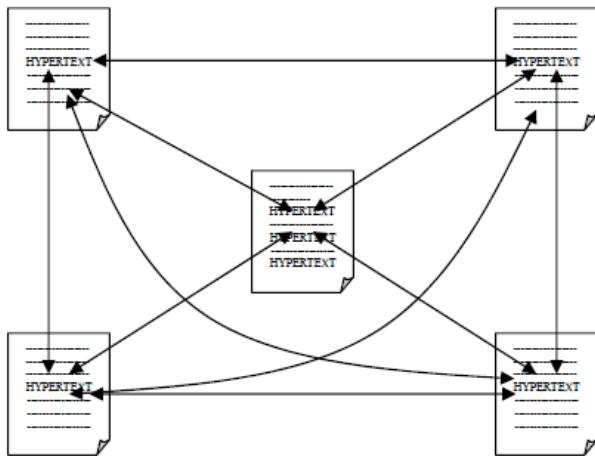
1.4 Pengantar Web

Webstite atau biasa disebut *web* adalah kumpulan dari halaman-halaman situs yang terangkum dalam sebuah *domain* atau *subdomain* yang berada di dalam *World Wide Web (WWW)* yang diakses melalui jaringan internet. *World Wide Web* atau *WWW* adalah gabungan atas semua situs yang dapat diakses publik. Sehingga *WWW* merupakan tempat pengaksesan dari semua situs *Web* yang ada.



Gambar 1.6 Ilustrasi Web

Sebuah halaman web biasanya berupa dokumen yang ditulis dalam format *HTML (Hyper Text Markup Language)* yang diakses melalui *HTTP (Hyper Text Transfer Protokol)*, yaitu sebuah protokol yang menyampaikan informasi dari *server website* untuk ditampilkan kepada para pemakai melalui *web browser*.



Gambar 1.6 Pengaksesan Informasi Melalui Hypertext

1.4.1 Sejarah Web

Awal perkembangan web dimulai pada bulan maret 1989 saat tim berner-lee yang bekerja di laboratorium fisika partikel eropa atau yang dikenal dengan nama *CERN (conseil european pour la recherche nuclaire)* yang terletak di genewa swiss, mengajukan protokol (bahasa atau prosedur yang digunakan untuk menghubungkan antara komputer yang satu dengan lainnya) sistem distribusi informasi internet yang digunakan untuk berbagai informasi di antara para fisikawan.

Protokol inilah yang selanjutnya dikenal sebagai protokol *world wide web* dan dikembangkan oleh *world wide web consortium (w3c)*. *w3c* adalah konsorsium dari sejumlah organisasi yang berkepentingan dalam pengembangan berbagai standar yang berkaitan dengan web.

HTTP (hypertext transfer protocol) merupakan protokol yang digunakan untuk mentransfer data antara *web server* ke *web browser*. Protokol ini mentransfer dokumen-dokumen web yang ditulis atau berformat *HTML (hypertext markup language)*. Dikatakan markup

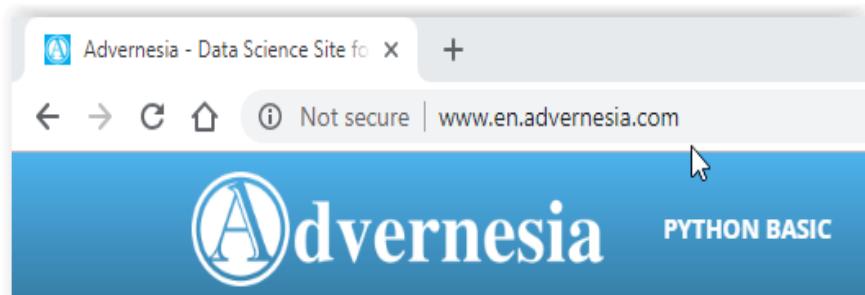
language karena *HTML* berfungsi untuk memperindah file teks biasa untuk ditampilkan pada program web browser. Hal ini dilakukan dengan menambahkan tag-tag (perintah khusus) pada file teks biasa tersebut.

1.4.2 *HTTP (Hypertext Transfer Protocol) dan HTTPS (Hyper Text Transfer Protocol Secure)*

1. *HTTP (Hypertext Transfer Protocol)*

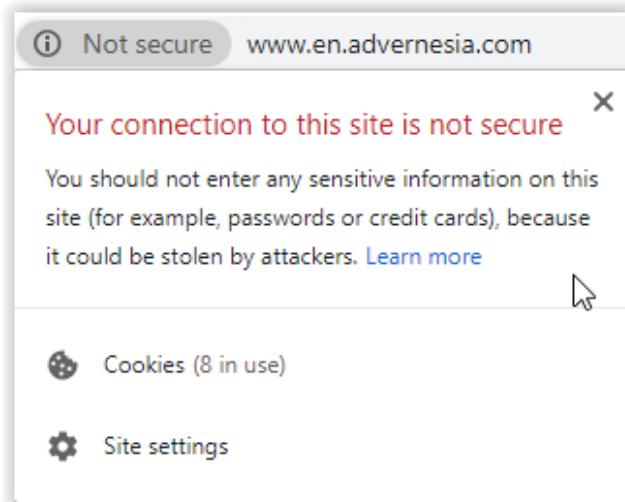
HTTP kepanjangan dari *Hyper Text Transfer Protocol*, adalah protokol standar yang digunakan sebuah website untuk melakukan transfer data antar komputer *server* (misalnya server hosting) dengan komputer *client* (komputer yang mengakses website). *HTTP* memegang peranan yang penting untuk mengatur aliran data dari komputer server terkait data apa saja yang akan diberikan kepada komputer *client* dan memberikan instruksi kepada komputer server untuk merespon komunikasi dari komputer *client*.

Semua website menggunakan protokol *HTTP*. Pada umumnya browser tidak menampilkan protokol *HTTP* yang digunakan, melainkan status koneksi website tersebut. *Browser* akan menampilkan status “*Not secure*” atau “**Tidak aman**” untuk protokol *HTTP*.



Gambar 1.7 Contoh situs http: Browser Google Chrome menampilkan status website dengan protokol HTTP

Status “*Not secure*” disebabkan karena pengelola website hanya menggunakan protokol *HTTP* saja tanpa memberikan perlindungan keamanan data pada website tersebut.



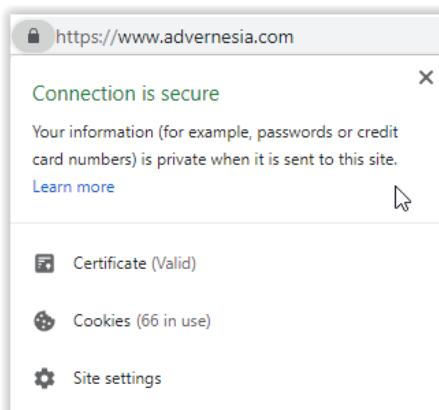
Gambar 1.8 Status “Not secure” pada browser Google Chrome

Status tersebut memperingati pengguna website untuk berhati-hati ketika memasukkan data pribadi, akun, password, hingga kartu kredit pada website bersangkutan. Hal ini disebabkan karena protokol *HTTP* sangat mudah untuk diretas.

2. *HTTPS (Hyper Text Transfer Protocol Secure)*

HTTPS kepanjangan *Hyper Text Transfer Protocol Secure*, merupakan protokol *HTTP* yang dilengkapi dengan sistem keamanan (security) berupa *SSL (Secure Socket Layer)*. Ibarat kabel telepon, *SSL* merupakan lapisan yang melindungi serat kabel. *HTTPS* memberikan perlindungan aliran data dari serangan peretasan. *HTTPS* dapat melindungi data website, privasi pengguna, akun bank online, akun pengguna website dari tindakan *cybercrime* seperti peretasan dan mengurangi resiko menipu pengguna dengan web tipuan (*phising*). Teknologi *HTTPS* sebenarnya sangat dibutuhkan oleh semua website, tidak hanya website yang dapat menyimpan data pengguna seperti toko online, website bank dan sosial media. Website kecil seperti blog pribadi dan profil perusahaan juga berpotensi dari peretasan.

Browser akan menampilkan status “**Connection is secure**” atau “**Koneksi aman**” saat mengakses suatu website yang menggunakan protokol *HTTPS*.



Gambar 1.9 Contoh situs https: Status “Secure” pada browser Google Chrome

Status “*Connection is secure*” memberikan informasi bahwa website tersebut dapat melindungi data yang dimasukkan pengguna website baik akun, password, atau kartu kredit.

1.4.3 *HTML (Hyper Text Markup Language)*

1. Pengenalan *HTML*

HTML (Hyper Text Markup Language) adalah merupakan sebuah dasar ataupun pondasi bahasa pemrograman sebuah webpage, HTML muncul sebagai standar baru dari kemajuan dan berkembangnya internet, pada pertama kali muncul internet masih dalam keadaan berbasis text dimana tampilan sebuah halaman web hanya berisikan sebuah text yang monoton tanpa sebuah format dokumen secara visual, bayangkan saja sebuah dokumen text yang dikemas dalam bungkus format seperti tipe file .txt atau sering disebut notepad, tanpa paragraph, satu warna, satu ukuran huruf tanpa gambar serta tidak adanya visual format dokumen seperti halnya Ms. Word, hal ini akan sangat membosankan dalam membaca. Dan selain itu pertama kali muncul internet user mengakses masih menggunakan sebuah terminal, hal itu jelas sangatlah tidak friendly. Pemrograman HTML muncul seiring perkembangan teknologi dan informasi.

2. Sejarah *HTML*

Hyper Text Markup Language (HTML) pertama kali diciptakan dan dikembangkan oleh Tim Berners-Lee pada awal tahun 1990-an yang pada saat itu masih bekerja di CERN. HTML diciptakan dengan tujuan sebagai cara sederhana namun efektif untuk mengkodekan dokumen elektronik. HTML pertama kali dipopulerkan dengan menggunakan browser Mosaic.

Tahun 1980, IBM memikirkan pembuatan suatu dokumen yang akan mengenali setiap elemen dari dokumen dengan suatu tanda tertentu. IBM kemudian mengembangkan suatu jenis bahasa yang menggabungkan teks dengan perintah-perintah pemformatan dokumen. Bahasa ini dinamakan Markup Language, sebuah bahasa yang menggunakan tanda-tanda sebagai basisnya. IBM menamakan sistemnya ini sebagai Generalized Markup Language atau GML.

Tahun 1986, ISO menyatakan bahwa IBM memiliki suatu konsep tentang dokumen yang sangat baik, dan kemudian mengeluarkan suatu publikasi (ISO 8879) yang menyatakan markup language sebagai standar untuk pembuatan dokumen-dokumen. ISO membuat bahasa ini dari GML milik IBM, tetapi memberinya nama lain, yaitu SGML (Standard Generalized Markup Language). ISO dalam publikasinya meyakini bahwa SGML akan sangat berguna untuk pemrosesan informasi teks dan sistem-sistem perkantoran. Tetapi diluar perkiraan ISO, SGML dan terutama subset dari SGML, yaitu HTML juga berguna untuk menjelajahi internet. Khususnya bagi mereka yang menggunakan World Wide Web.

Mulai pada tahun 1989, sebuah nama HTML muncul dari pemikiran Caillau Tim yang bekerja sama dengan Banners Lee Robert yang ketika itu masih bekerja di CERN memulai mengembangkan bahasa pemrograman ini, dan dipopulerkan pertama kali dengan browser Mosaic. Dan mulailah dari tahun 1990 HTML sangat berkembang dengan cepat hingga mencapai versi HTML versi 5.0 yang digarap pada 4 Maret 2010 kemarin oleh W3C.

Sejarah dari standar HTML:

1. HTML 2.0 (RFC 1866) disetujui sebagai standar 22 September 1995

2. HTML 3.2 14 Januari 1996
3. HTML 4.0 18 Desember 1997
4. HTML 4.01 (minor Fixes) 24 Desember 1999
5. ISO/IEC 15445:2000 (“ISO HTML”, berdasar pada HTML 4.01 Strict) 15 Mei 2000
6. HTML 5 masih dalam draft pengerjaan Januari 2008

3. Versi *HTML*

a. *HTML* 1.0

Ini adalah awal mula dari HTML (pendahulunya). Pada versi ini masih terlihat beberapa kelemahan dan masih sangat sederhana. Kemampuan yang dimiliki oleh versi 1.0 ini hanya terbatas pada heading, paragraph, hypertext, list, dan setak tebal atau miring pada teks.

b. *HTML* 2.0

Versi 2.0 pada 14 Januari 1996, pada versi ini ada beberapa tambahan kemampuan diantaranya penambahan form comment, hal ini menyebabkan adanya sebuah interaktif dan mulai dari versi ini yang menjadikan sebuah pioneer dalam perkembangan homepage interaktif.

c. *HTML* 3.0

Dirilis pada 18 Desember 1997 yang sering disebut sebagai HTML+ yang mempunyai kemampuan dalam beberapa fasilitas diantaranya adalah penambahan fitur table dalam paragraph, akan tetapi versi ini tidak bertahan lama.

d. *HTML* 3.2

Dan pada bulan Mei 1996 dikeluarkan versi baru sebagai pengganti dan penyempurnaan versi 3.0 ini yaitu HTML veri 3.2,

keluarnya versi ini dikarenakan adanya beberapa kasus yang timbul pada pengembang browser yang telah melakukan pendekatan dengan cara lain yang justru hal tersebut menjadi popular, maka dibakukan versi 3.2 untuk mengakomodasi praktek yang banyak digunakan oleh pengembang browser dan diterima secara umum, dapat dikatakan bahwa versi 3.2 ini merupakan versi 3.0 yang dikembangkan oleh beberapa pengembang browser seperti Netscape dan Microsoft.

e. *HTML 4.0*

Yang terakhir perombakan terjadi pada tahun 1999 tepatnya tanggal 24 Desember yaitu HTML versi 4.0, seperti yang kita kenal HTML pada saat ini penambahan link, meta, imagemaps. Image dan lain-lain sebagai penyempurnaan versi 3.2. Di samping itu versi ini ditambahkan tag-tag baru seperti ABBR, ACRONYM, BUTTON, PARAM, BUTTON, TBODY, THEAD dan lain sebagainya.

f. *HTML 5.0*

Pada tanggal 4 Maret 2010, terdapat sebuah informasi bahwasannya HTML versi 5.0 masih dikembangkan oleh W3C (World Wide Web Consortium) dan IETF (Internet Engineering Task Force) yaitu sebuah organisasi yang menangani HTML sejak versi 2.0.

4. Struktur Dasar *HTML*

Struktur dasar dokumen HTML adalah sebagai berikut :

```
<html>
<head>
<title>Disini Judul Dokumen HTML</title>
```

```
</head>
```

```
<body>
```

Disini penulisan informasi Web

```
</body>
```

```
</html>
```

Dari struktur dasar HTML di atas dapat dijelaskan sebagai berikut:

a. Tag

Adalah teks khusus (*markup*) berupa dua karakter "<" dan ">", sebagai contoh **<body>** adalah tag dengan nama body. Secara umum tag ditulis secara berpasangan, yang terdiri atas **tag pembuka** dan **tag penutup** (ditambahkan karakter "/" setelah karakter "<"), sebagai contoh **<body>** ini adalah tag pembuka isi dokumen HTML, dan **</body>** ini adalah tag penutup isi dokumen HTML.

b. Element

Element terdiri atas tiga bagian, yaitu tag pembuka, isi, dan tag penutup. Sebagai contoh untuk menampilkan judul dokumen HTML pada web browser digunakan element title, dimana:

ini adalah tag penutup judul dokumen HTML. Tag-tag yang ditulis secara berpasangan pada suatu element HTML, tidak boleh saling tumpang tindih dengan pasangan tag-tag lainnya.

Contoh penulisan tag-tag yang benar

<p>

.....

```
</b>  
</p>
```

Contoh penulisan tag-tag yang salah

```
<p>  
<b>  
.....  
</p>  
</b>
```

c. Attribute

Attribute mendefinisikan property dari suatu element HTML, yang terdiri atas nama dan nilai. Penulisannya adalah sebagai berikut:

```
<TAG>  
nama-attr="nilai-attr"  
nama-attr="nilai-attr"  
.....  
>  
.....  
</TAG>
```

Secara umum nilai attribute harus berada dalam tanda petik satu atau dua.

d. Element HTML

Menyatakan pada browser bahwa dokumen Web yang digunakan adalah HTML.

Sintaks:

<html>

.....

</html>

e. Element HEAD

Merupakan kepala dari dokumen HTML. Tag dan tag terletak di antara tag dan tag.

Sintaks:

<head>

.....

</head>

f. Element Title

Merupakan judul dari dokumen HTML yang ditampilkan pada judul jendela browser. Tag **<title>** dan tag **</title>** terletak di antara tag **<head>** dan tag **</head>**.

Sintaks:

<title>

.....

</title>

g. Element Body

Element ini untuk menampilkan isi dokumen HTML. Tag **<body>** dan tag **</body>** terletak di bawah tag **<head>** dan tag **</head>**. Element BODY mempunyai attribute-attribute yang menspesifikasikan khususnya warna dan latarbelakang dokumen yang akan ditampilkan pada browser.

Sintaks:

<body text="v" bgcolor="w" background="uri" link="x" alink="y" vlink="z">

.....

</body>

Attribute text memberikan warna pada teks, bgcolor memberikan warna pada latarbelakang dokumen HTML, background memberikan latar belakang dokumen HTML dalam bentuk gambar, link memberikan nilai warna untuk link, alink memberikan warna untuk link yang sedang aktif, vlink memberikan warna untuk link yang telah dikunjungi. Jika attribute bgcolor dan background keduanya dispesifikasikan maka attribute background yang akan digunakan, akan tetapi jika nilai attribute background (gambar) tidak ditemukan pada dokumen HTML maka attribute bgcolor yang akan digunakan.

5. Kelebihan dan Kekurangan HTML

Kelebihan *HTML*:

- a. Merupakan bahasa pengkodean yang lintas platform (cross platform), maksudnya HTML dapat digunakan pada berbagai jenis mesin komputer yang berbeda dan berbagai macam sistem operasi yang berbeda. Jadi berdifikat fleksibel karena ditulis cukup dengan menggunakan editor karakter ASCII.
- b. Dapat disisipi gambar baik gambar statis atau dinamis (animasi) termasuk menggunakan gambar untuk dijadikan hyperlink. Gambar disini digunakan untuk merujuk pada suatu halaman web, dimana setiap titik-titik yang sudah didefinisikan berupa rectangular (kotak), poligon (kurva tak beraturan) atau lingkaran digunakan untuk ‘jump’ ke halaman lain, atau link ke halaman di luar web yang bersangkutan.

- c. Dapat disisipi animasi berupa Java Applet atau file-file animasi dari Macromedia Flash atau Macromedia Shockwave (untuk keperluan ini, browser harus memiliki plug-in khusus untuk menjalankan file-file animasi ini).
- d. Dapat disisipi bahasa pemrograman untuk mempercantik halaman web seperti Javascript, Vbscript, Active Server Pages, Perl, Tcl, PHP, dan sebagainya.
- e. Bukan merupakan bahasa pemrograman jadi tidak memerlukan kompiler. Cara menjalankannya cukup dengan menggunakan browser.

Kekurangan *HTML*:

- a. Menghasilkan halaman yang statis, untuk memperoleh halaman yang dinamis harus menggunakan bahasa pemrograman tertentu seperti Javascript atau Vbscript dan animasi seperti Flash atau Shockwave.
- b. Memiliki tag-tag yang begitu banyak sehingga susah dipelajari untuk yang masih awam.
- c. Tidak dapat menghasilkan halaman yang interaktif. Interaktif disini maksudnya client dapat berinteraksi dengan server. Untuk keperluan itu HTML harus disisipi bahasa pemrograman yang dapat menangani hal tersebut, contohnya Perl dan Tcl.

1.4.4 Hubungan *HTML* dan *PHP*

Halaman web biasanya disusun dari kode-kode html yang disimpan dalam sebuah file berekstensi .html. File html ini dikirimkan oleh server (atau file) ke browser, Kemudian browser menerjemahkan kode-kode tersebut sehingga menghasilkan suatu tampilan yang indah. Lain halnya dengan program php, program ini

harus diterjemahkan oleh web-server sehingga menghasilkan kode html yang dikirim ke browser agar dapat ditampilkan. Program ini dapat berdiri sendiri ataupun disisipkan di antara kode-kode html sehingga dapat langsung ditampilkan bersama dengan kode-kode html tersebut. Program php dapat ditambahkan dengan mengapit program tersebut diantara tanda.

Tanda-tanda tersebut biasanya disebut tanda untuk escaping (kabur) dari kode html. File html yang telah dibubuhi program php harus diganti ekstensi-nya menjadi .php3 atau php. Php merupakan bahasa pemograman web yang bersifat server-side HTML=embedded scripting, di mana script-nya menyatu dengan HTML dan berada si server. Artinya adalah sintaks dan perintah-perintah yang kita berikan akan sepenuhnya dijalankan di server tetapi disertakan HTML biasa. PHP dikenal sebagai bahasa scripting yang menyatu dengan tag HTML, dieksekusi di server dan digunakan untuk membuat halaman web yang dinamis seperti ASP (Active Server Pages) dan JSP (Java Server Pages).

1.4.5 Pemahaman Pemrograman Web

1. Pemrograman: Suatu usaha menulis sesuatu perintah (program aplikasi) sehingga computer dapat menjalankan apa yang kita inginkan.
2. Pemrograman Web: Membuat program aplikasi berbasis web
3. Aplikasi berbasis web: Aplikasi yang dibuat dengan memanfaatkan mekanisme dan aplikasi yang sudah ada pada sistem web (WWW).

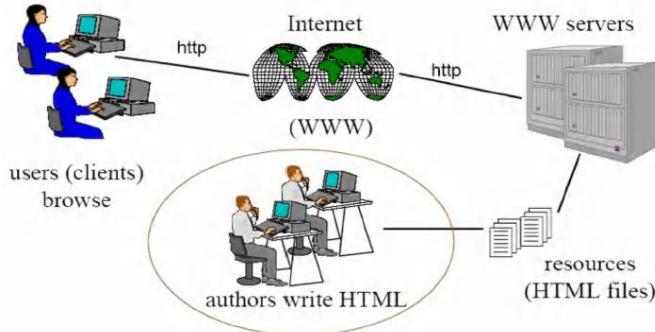
1.4.6 Membuat Aplikasi Berbasis Web

1. Memperkaya fungsi web server dengan cara menambahkan program pada dokumen yang akan dieksekusi oleh server ketika file dokumen web tersebut diakses oleh web server.

Misalnya: Program yang mengambil data ke basis data untuk ditampilkan ke web browser.

2. Memperkaya interaktivitas dokumen dengan cara menambahkan program pada dokumen yang akan dieksekusi oleh web browser ketika file dokumen tersebut ditampilkan oleh web browser.

Misalnya: Program yang memvalidasi data masukan pada form sebelum disubmit ke web server.



Gambar 1.10 Cara Kerja Web

1.4.7 Keunggulan Dan Kekurangan Aplikasi Berbasis Web

1. Keunggulan

- Dapat diakses kapanpun dan dari mana pun selama ada koneksi internet.
- Dapat diakses hanya dengan menggunakan browser (umumnya sudah tersedia di PC, PDA, dan handphone).
- Tidak perlu menginstal aplikasi client khusus.

2. Kekurangan

- a. Antarmuka yang dibuat terbatas sesuai spesifikasi standar untuk membuat dokumen web.
- b. Keterbatasan kemampuan web browser untuk menampilkannya.
- c. Terbatasnya kecepatan internet mungkin membuat respon aplikasi menjadi lebih lambat.

1.4.8 Web Statis dan Web Dinamis

1. Web Statis

Web statis ialah web yang berisi tentang informasi yang memiki sifat statis (tetap) atau pengguna tidak dapat berinteraksi dengan website tersebut, web statis dapat dilihat dari tampilan website tersebut jika suatu web hanya berhubungan dengan halaman web lain yang berisi informasi tetap maka web tersebut termasuk kedalam kategori web statis, pada web statis pengguna hanya dapat melihat isi web tersebut dan jika di klik hanya akan berpindah pada halaman lainnya. Dalam web statis interaksi pengguna sangatlah terbatas

2. Web Dinamis

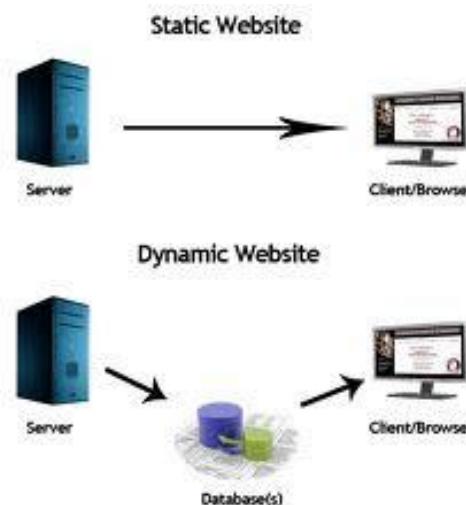
Web dinamis adalah web yang dapat menampilkan informasi serta dapat membuat pengguna berinteraksi seperti dengan form input, button sehingga dapat mengolah informasi yang ditampilkan pada web tersebut, web dinamis bersifat tidak kaku dan terlihat lebih enak dipandang.

3. Perbedaan Web Statis dan Web Dinamis

a. Web Statis

- 1. Sebagian besar halaman web statis.

2. Isi (teks/link/gambar) yang sama setiap kali diakses *HyperText Markup Language (HTML)* digunakan untuk menetukan teks/ format gambar.
 3. Contoh: dokumen online, kebanyakan homepage.
- b. Web Dinamis
1. Sebagai web dinamis mengarah ke layanan online/ e-commerce.
 2. Halaman webnya juga harus menyediakan konten dinamis.
 3. Halaman harus update, berubah-ubah (misalnya: berputar banner, artikel ganti)
 4. Harus mampu bereaksi terhadap tindakan info pengguna, permintaan dan proses, pemesanan jasa, dan lain-lain.



Gambar 1.11 Web Statis vs Web Dinamis

1.4.9 Basis Data

1. Pengertian Basis Data

Basis Data terdiri dari kata basis dan data. Basis dapat diartikan sebagai markas atau gudang. Sedangkan data adalah catatan atas

kumpulan fakta dunia nyata yang mewakili objek seperti manusia, barang, hewan, konsep, peristiwa dan sebagainya yang diwujudkan dalam bentuk huruf, angka, simbol, gambar, teks, bunyi atau kombinasinya.

Sebagai suatu kesatuan maka pengertian basis data atau biasa disebut *database* adalah sebagai berikut:

Pengertian Basis Data atau *Database*:

1. Himpunan kelompok data yang saling terhubung dan diorganisasi sedemikian rupa supaya kelak dapat dimanfaatkan kembali secara cepat dan mudah.
2. Kumpulan data dalam bentuk file/tabel/arsip yang saling berhubungan dan tersimpan dalam media penyimpanan elektronis, untuk kemudahan dalam pengaturan, pemilahan, pengelompokan dan pengorganisasian data sesuai tujuan.

Dengan basis data seseorang dapat menyimpan sebuah informasi, seperti data mahasiswa, kepegawaian atau produk ke dalam media penyimpanan elektronis seperti cakram magnetis (*disk*) melalui perangkat komputer, Untuk kemudian data tersebut dapat kita gunakan sesuai keperluan. Database mempunyai 8 operasi dasar diantaranya adalah *Create database, Drop database, create table, Drop table, Insert, Read, Update dan Delete*.

2. Pengertian Sistem Basis Data

Sistem basis data adalah sistem yang terdiri atas kumpulan tabel data yang saling berhubungan dan kumpulan program yang memungkinkan beberapa pemakai atau program lain untuk mengakses dan memanipulasi tabel tabel data tersebut.

3. Komponen Sistem Basis Data

a. Perangkat Keras

Perangkat keras atau hardware yang umumnya terdapat dalam sistem basis data adalah komputer, hard disk, memori sekunder offline (removable disk, fd), perangkat komunikasi jaringan.

b. Sistem Operasi

Sistem operasi adalah program yang dirancang untuk mengaktifkan sistem komputer dan mengendalikan seluruh sumber daya yang ada di dalamnya termasuk operasi- operasi dasar komputer. seperti Windows, Unix dan Linux.

c. Basis Data

Komponen adalah sekumpulan data yang terorganisir dengan baik sehingga data tersebut mudah disimpan, diakses, dan juga dapat dimanipulasi. Sistem basis data dapat terdiri dari beberapa basis data yang memiliki data masing- masing.

d. Database Management System (DBMS)

DBMS atau database management system adalah program aplikasi khusus yang dirancang untuk membuat dan juga mengelola database yang tersedia. Sistem ini berisi koleksi data dan set program yang digunakan untuk mengakses database tersebut.

DBMS adalah software yang berperan dalam mengelola, menyimpan, dan mengambil data kembali. Adapun mekanisme yang digunakan sebagai pelengkap adalah pengaman data, konsistensi data dan pengguna data bersama.

Contoh dari DBMS adalah *Microsoft Access, MySql, Oracle database, Sybase, Borland-Interbase, PostgreSQL dll.*

e. Pemakai atau User

User adalah salah satu komponen database yang berinteraksi secara langsung dengan database. Ada beberapa tipe user, diantaranya, programmer aplikasi, User mahir (casual user), user umum (end user) dan user khusus (specialized user)

f. Aplikasi atau Perangkat Lain

Aplikasi ini tergantung kebutuhan, pemakai basis data bisa dibuatkan program khusus untuk melakukan pengisian, pengubahan atau pengambilan data yang mudah dalam pemakaianya. Program tersebut ada yang tersedia langsung dalam DBMS atau dibuat menggunakan aplikasi lain seperti misalnya Visual Basic.

4. Bahasa Basis Data (Database Language)

Bahasa database merupakan bahasa data yang dapat ditempelkan kedalam bahasa pemrograman yang lain, sebut saja Java, Pascal, Fortran dst. Bahasa dimana instruksi data base menempel disebut inang. Beberapa komponen Bahasa data base menurut fungsinya dibagi menjadi:

a. *Data Definition Language*

Data definition language adalah sekumpulan definisi yang disimpan di dalam data dictionary.

b. *Data Manipulation Language*

Data Manipulation Language berisi akumulasi dari operasi manipulasi basis data yang dilakukan. Ini biasa disebut dengan bahasa query sebab biasanya digunakan untuk meminta informasi yang ada dari basis data tersebut.

5. Fungsi dan Tujuan Basis Data

Fungsi basis data cukup banyak dan cakupannya pun luas dalam mendukung keberadaan lembaga atau organisasi, diantaranya adalah:

a. Ketersediaan/ *Availability*

Fungsi basis data yang pertama adalah untuk menyediakan data-data penting saat sedang diperlukan. Ya, ini adalah fungsi penting dari basis data yang meskipun tidak terletak dalam satu lokasi, dan tersimpan dalam bentuk disk, akan tetapi dengan cara penyimpanan yang sistematik, informasi tersebut mudah untuk didapatkan.

b. Mudah dan Cepat/ *Speed*

Selanjutnya, fungsi dari basis data ini adalah agar Anda sebagai pengguna bisa dengan mudah mengaksesnya saat sedang membutuhkan. Tidak perlu tunggu nanti, apalagi harus mengalokasikan waktu tertentu untuk memanggilnya.

c. Kelengkapan/ *Completeness*

Basis data harus menyimpan data yang lengkap, yang bisa melayani keperluan penggunanya secara keseluruhan. Meski kata lengkap yang dipakai disini sifatnya relatif, namun setidaknya data tersebut membantu memudahkan untuk menambah koleksi data, dan menjamin mudahnya pengguna untuk memodifikasi struktur data yang ada, sebut saja field-field data yang tersedia.

d. *Accuracy* dan *Security*

Fungsi data base selanjutnya adalah untuk accuracy atau keakuratan. Jadi, agar kesalahan dapat ditekan semaksimal

mungkin, Anda bisa lakukan pengorganisasian file-file database dengan baik untuk menghindari kesalahan pada proses data entry dan juga dalam proses penyimpanan atau datastore.

Selain itu, fungsi database adalah untuk security atau keamanan. Ada fasilitas pengaman data yang disediakan oleh sistem basis data yang baik sehingga data tidak bisa dimodifikasi, diakses, diubah maupun dihapus oleh yang tidak mendapatkan hak untuk melakukannya.

e. *Storage Efficiency*

Pengorganisasian data dilakukan dengan baik dengan tujuan untuk menghindari duplikasi data yang berpengaruh pada bertambahnya ruang penyimpanan dari basis data tersebut. pengkodean dan juga relasi data bermanfaat untuk menghemat space penyimpanan dalam basis data.

1.5 Pengantar *Approval*

Definisi *approval* menurut KBBI adalah kata dalam bahasa inggris yang artinya persetujuan, penerimaan, restu dan izin. *Approval* masuk ke dalam bahasa inggris atau *English* yaitu bahasa jermanik yang pertama kali dituturkan di Inggris pada abad pertengahan awal dan saat ini merupakan bahasa yang paling umum digunakan di seluruh dunia.

1.5.1 Sistem Informasi *Approval*

Sistem informasi *approval* merupakan sebuah sistem informasi yang bertujuan untuk melakukan persetujuan terlebih dahulu sebelum data dapat di proses lebih lanjut.

BAB II

PENGENALAN *FRAMEWORK, CODEIGNITER, E-MAIL* DAN PENJELASAN *TOOLS* YANG DIGUNAKAN

Framework codeigniter dan notifikasi *e-mail* merupakan dua fitur yang akan digunakan dalam pembuatan sistem informasi *approval* ini serta ada beberapa *tools* lainnya yang. Pada bab ini penulis akan memaparkan dua hal ini secara rinci.

2.1 *Framework*

Sebelum masuk ke *framework codeigniter* mari kita bahas dulu apa itu *framework*, beserta fungsi dan jenis-jenisnya.

2.1.1 Pengenalan *Framework*

Apa itu *framework*? *Framework* secara sederhana disebut kerangka kerja. *Developer* menggunakan *framework* untuk memudahkan mereka dalam membuat dan mengembangkan aplikasi atau *software*. Pada *framework* itu sendiri berisi kumpulan fungsi-fungsi dasar atau perintah yang biasa digunakan dalam mengembangkan suatu *software*, dengan tujuan agar *software* yang dibangun menjadi lebih cepat dan terstruktur.

Jadi *framework* adalah sebuah *software* untuk memudahkan para *programmer* untuk membuat sebuah aplikasi *web* yang di dalamnya ada berbagai fungsi diantaranya *plugin*, konsep untuk membentuk sebuah sistem tertentu agar tersusun dan tersuktur dengan rapih.

2.1.2 Jenis-Jenis *Framework*

1. *Laravel*

Framework ini diperkenalkan pada tahun 2011, *Laravel* kini telah menjadi *framework PHP open-source* gratis paling populer di

dunia. Kok bisa? Karena *framework* ini dapat menangani aplikasi *web* yang kompleks dengan aman, dengan kecepatan yang jauh lebih cepat dari pada *framework* lainnya. *Laravel* menyederhanakan proses pengembangan dengan memudahkan tugas-tugas umum seperti *routing*, *session*, *caching* dan otentikasi.



Gambar 2.1 Ilustrasi Framework Laravel

2. *CodeIgniter*

Dikenal karena ukurannya yang kecil (hanya berukuran sekitar 2 MB, termasuk dokumentasinya) *CodeIgniter* adalah *framework PHP* yang cocok untuk mengembangkan situs *web* dinamis. Ia menawarkan banyak modul prebuilt yang membantu membangun komponen yang kuat dan dapat digunakan kembali.

3. *Symfony*

Framework Symfony ini diluncurkan pada tahun 2005, dan meskipun sudah ada lebih lama dari pada *framework* lainnya yang ada dalam daftar ini, *symfony* adalah *platform* yang handal dan matang. *Symfony* adalah *framework PHP MVC* yang luas dan satu-satunya *framework* yang diketahui mengikuti standar *PHP* dan *web*.



Gambar 2.2 Ilustrasi Framework Symfony

4. *CakePHP*

Jika anda mencari *toolkit* yang sederhana dan elegan, sekarang tidak perlu lagi mencarinya. *CakePHP* akan membantu Anda mengembangkan situs web yang mengesankan secara visual dan fitur. Selain itu, *CakePHP* adalah salah satu framework termudah untuk dipelajari, terutama karena kerangka *CRUD*-nya (*creat, read, update, dan delete*). *CakePHP* memasuki pasar pada awal 2000-an, dan sejak saat itu memperoleh kinerja yang lebih baik dan banyak terus memiliki komponen terbaru.



Gambar 2.3 Ilustrasi Framework CakePHP

5. *Yii*

Framework Yii Ini adalah *framework PHP* berbasis komponen dan kinerja tinggi untuk mengembangkan aplikasi web modern. *Yii* cocok untuk semua jenis aplikasi *web*. *Yii* menyediakan fitur yang

aman dan profesional untuk membuat proyek yang kuat dengan cepat.



Gambar 2.4 Ilustrasi Framework Yii

6. Zend Framework

Framework Zend adalah *framework open source* yang berorientasi objek cukup lengkap. *Framework* ini dibangun pada metodologi *MVC*, yang membantu Anda untuk menghasilkan aplikasi berkualitas tinggi kepada klien perusahaan. *Framework Zend* berisi kumpulan paket *PHP* yang dapat digunakan untuk mengembangkan aplikasi dan layanan *web*.



Gambar 2.5 Ilustrasi Framework ZendFramework

7. Phalcon

Merupakan *framework PHP full-stack* yang menggunakan pola desain arsitektur *web MVC*, *Phalcon* pada awalnya ditulis dalam C dan C ++ dan dirilis pada 2012. Karena *framework* ini dikirim sebagai C-extension, Anda tidak perlu khawatir belajar bahasa pemrograman C.



Gambar 2.6 Ilustrasi Framework Phalcon

8. *FuelPHP*

FuelPHP adalah *framework PHP full-stack* yang fleksibel yang pertama kali dirilis pada 2011. Selain mendukung pola desain *MVC*, ia memiliki versi sendiri yang disebut *hierarchical model view controller (HMVC)*. Dengan *HMVC*, tidak seperti dengan *MVC*, konten tidak perlu digandakan untuk ditampilkan di beberapa halaman. Akibatnya, ia menghabiskan lebih sedikit waktu dan memori.



Gambar 2.7 Ilustrasi Framework FuelPHP

9. *PHPixie*

Diperkenalkan pada tahun 2012 dan seperti halnya *FuelPHP*, *PHPixie* mengimplementasikan pola desain *HMVC*. Tujuannya adalah untuk menciptakan *framework* berkinerja tinggi untuk aplikasi berbasis *web*.



Gambar 2.8 Ilustrasi Framework PHPixie

10. *Slim*

Slim adalah framework PHP yang cukup populer yang dapat membantu pengembang membuat aplikasi web dan API yang sederhana namun powerfull.



Gambar 2.9 Ilustrasi Framework Slim

2.2 Framework *CodeIgniter*



Gambar 2.10 Ilustrasi Framework CodeIgniter

2.2.1 Pengenalan *Framework CodeIgniter*

Dalam situs resmi *codeigniter* menyebutkan bahwa *codeigniter* merupakan *framework PHP* yang kuat dan sedikit bug. *CodeIgniter* ini dibangun untuk para *developer* dengan bahasa pemrograman *PHP* yang membutuhkan alat untuk membuat sebuah situs *website* dengan fitur yang lengkap. *Framework codeigniter* dikembangkan oleh Rick Ellish, CEO Ellishlab, Inc.

CodeIgniter merupakan *framework PHP* yang dibuat berdasarkan *MVC (Model View Controller)*. Dimana *MVC* merupakan sebuah pola desain (*desaign pattern*) arsitektur pengembangan aplikasi yang memisahkan dan mengelompokkan beberapa kode sesuai dengan fungsinya *MVC* membagi aplikasi je dalam tiga bagian fungsional: *model*, *view* dan *controller*.

2.2.2 Bahasa Pemrograman *PHP*



Gambar 2.11 Ilustrasi Bahasa Pemrograman *PHP*

2.2.2.1 Pengenalan *PHP*

PHP atau kependekan dari *Hypertext Preprocesor* adalah salah satu bahasa pemrograman *open source* yang sangat cocok atau dikhkususkan untuk pengembangan *web* dan dapat ditanamkan pada sebuah skripsi *HTML*. Bahasa *PHP* dapat

dikatakan menggambarkan beberapa bahasa pemograman seperti *C*, *Java*, dan *Perl* serta mudah dipelajari.

PHP merupakan bahasa *scripting server – side*, dimana pemrosesan datanya dilakukan pada sisi *server*. Sederhananya, *serverlah* yang akan menerjemahkan skrip program, baru kemudian hasilnya akan dikirim kepada *client* yang melakukan permintaan.

Sistem kerja dari *PHP* diawali dengan permintaan yang berasal dari halaman *website* oleh *browser*. Berdasarkan *URL* atau alamat *website* dalam jaringan internet, *browser* akan menemukan sebuah alamat dari *webserver*, mengidentifikasi halaman yang dikehendaki, dan menyampaikan segala informasi yang dibutuhkan oleh *webserver*. Selanjutnya *webserver* akan mencarikan berkas yang diminta dan menampilkan isinya di *browser*. *Browser* yang mendapatkan isinya segera menerjemahkan kode *HTML* dan menampilkannya. Lalu bagaimana apabila yang dipanggil oleh *user* adalah halaman yang mengandung *script PHP*? Pada prinsipnya sama dengan memanggil kode *HTML*, namun pada saat permintaan dikirim ke *web-server*, *web-server* akan memeriksa tipe *file* yang diminta *user*. Jika tipe *file* yang diminta adalah *PHP*, maka akan memeriksa isi *script* dari halaman *PHP* tersebut.

Apabila dalam *file* tersebut tidak mengandung *script PHP*, permintaan *user* akan langsung ditampilkan ke *browser*, namun jika dalam *file* tersebut mengandung *script PHP*, maka proses akan dilanjutkan ke modul *PHP* sebagai mesin yang menerjemahkan *script-script PHP* dan mengolah *script*

tersebut, sehingga dapat dikonversikan ke kode-kode *HTML* lalu ditampilkan ke *browser user*.

2.2.2.2 Kelebihan dan Kekurangan *PHP*

1. Kelebihan *PHP*

a. Memiliki *Community* yang besar

Programmer *Web* mana yang tidak mengetahui *PHP*, semua *web programmer* paling tidak pasti pernah mencoba *PHP*. Banyak sekali website yang menggunakan *PHP* sebagai bahasa pemrograman untuk membuat aplikasi *web* atau *website* nya. *Facebook*, *Yahoo*, *Wikipedia*, *WordPress* adalah contoh website terkenal yang menggunakan *PHP*. Forum untuk membahas dan juga saling bertukar pikiran dalam pemrograman *PHP* juga telah banyak muncul di berbagai situs. Kebanyakan kuliah di bidang *IT* mengajarkan *PHP* sebagai bahasa pemrograman awal untuk mahasiswanya yang berkuliah di jurusan *website development*.

b. Mudah Dipelajari

PHP mudah di install dan dikonfigurasi. membuatnya menjadi bahasa pemrograman tingkat *entry level* yang mudah dipelajari bagi seseorang yang baru memulai belajar pengembangan *web*. Tutorial untuk memulai belajar pemrograman *PHP* dapat diperoleh dengan mudah secara *online*, di toko buku, ataupun di lembaga bimbingan kursus pengembangan *website*.

c. Pengembangan Cepat

Membuat Aplikasi menggunakan *PHP* jauh lebih cepat daripada mengembangkan aplikasi *web* menggunakan

bahasa pemrograman lain. banyak sekali *tools*, *boiler* yang tersedia secara *open source* untuk bahasa pemrograman *PHP*. hal ini mempercepat proses dari *start* sampai dengan *finish* sebuah projek pembuatan aplikasi web.

d. *Ringkas*

Bagi *Programmer web* yang pernah mencoba bahasa *ASP* maupun *java* pasti mengetahui betul satu kelebihan ini. Mulai dari proses *install* yang tidak perlu setting berlebihan, konfigurasi dengan *database* yang mudah, hingga proses pengembangan yang tidak memerlukan waktu kompilasi. membuat *PHP* terasa sangat ringkas dan praktis berbeda dengan bahasa pemrograman lain yang membutuhkan proses kompilasi untuk dapat melihat website yang telah diselesaikan pembuatan kodennya. Bahkan, bahasa pemrograman *php* dapat digunakan didalam dokumen *html*.

e. *Maintenance* mudah

Sekali web yang menggunakan *PHP* berjalan, *programmer* dapat dengan mudah melakukan *update* dari *software PHP* dengan mudah jika memang diperlukan. karena sifat *PHP* yang merupakan *interpreter*. Aplikasi *web* yang dibuat dengan menggunakan *PHP* dapat dengan mudah di *upgrade* versi *PHP* tanpa harus melakukan kompilasi ulang *source code*. Berbeda sekali dengan bahasa pemrograman lain yang membutuhkan kompilasi ulang jika melakukan upgrade versi dari bahasa pemrograman. *PHP* juga dapat berjalan pada berbagai macam *web server* seperti *apache*, *nginx*, dan *IIS*.

f. Open Source

PHP merupakan sebuah projek *open source* dengan *license* yang dikeluarkan oleh *PHP group* yaitu *PHP license* V3.01. Inti dari *license* ini adalah setiap pengguna program *PHP* bebas menggunakan *PHP* secara gratis tanpa harus memberikan royalty apapun ke *PHP group* namun tetap wajib mencantumkan licensi atas *PHP* yang dimiliki *PHP Group*. Dengan kata lain selama pemakai program *PHP* tidak mengakui produk *PHP* adalah buatannya maka perjual belian program yang menggunakan *PHP* diperbolehkan tanpa harus membayar *licensi* apapun.

g. Perkembangan Pesat

Karena sifat *PHP* yang *open source*, banyak sekali bermunculan projek projek *open source* besar yang menggunakan *PHP* seperti *Prestashop*, *WordPress*, *Drupal*, dan lain lain. Hal ini menjadi keunggulan yang sangat besar bagi orang yang menguasai pemrograman *PHP*. Dengan sangat luasnya perkembangan *PHP*, maka kesempatan untuk bisnis ataupun kerja pada bidang pemrograman *PHP* sangatlah luas

2. Kekurangan *PHP*

a. Banyak kompetisi

Komunitas yang banyak tentu membawa kompetisi yang ketat. Para *web developer* yang menguasai *PHP* tiap hari semakin bertambah. Namun kekurangan ini seharusnya menjadi pemacu bagi para pebisnis yang menginginkan produk *IT* untuk menggunakan *PHP* sebagai bahasa

pemrograman yang digunakan untuk mengembangkan aplikasi bisnisnya karena terbukanya para programmer *PHP* yang sangat kompetitif dan tiap hari semakin banyak

b. Mudah di bajak

Karena sifat *PHP* yang merupakan *interpreter, source code* dari aplikasi *php* dapat dengan mudah di modifikasi dan diubah fungsinya. hal ini membuat *PHP* tidak cocok untuk digunakan mengembangkan aplikasi jika pemilik aplikasi memiliki *source code* yang ingin dijaga kerahasiaannya.

Meskipun ada cara untuk mengamankan *source code* yang menggunakan bahasa pemrograman *PHP*, namun dibutuhkan sebuah extensi yang berbayar yang dikeluarkan oleh *Zend* sebuah corporasi di bidang pemrograman *PHP*

c. Terkesan kurang prestigious

Entry level yang berada pada tingkat pemula, yakni mudah dipelajari oleh programmer pemula membuat bahasa pemrograman *PHP* terkesan kurang prestisius jika dibandingkan dengan bahasa pemrograman web lain yang terkesan lebih sulit untuk digunakan. Aplikasi web yang dihasilkan dari penggunaan bahasa pemrograman *PHP* terkesan kurang aman dan memiliki celah. Namun sebenarnya hal ini disebabkan karena faktor pengembang yang mungkin belum mempelajari secara penuh bagaimana standar dan cara membuat aplikasi yang benar dengan menggunakan *PHP*. Dari sisi performa, pemrograman *PHP* dan pemrograman lainnya jika

digunakan dengan standar dan penerapan yang benar, akan menghasilkan sebuah aplikasi *web* yang berkualitas

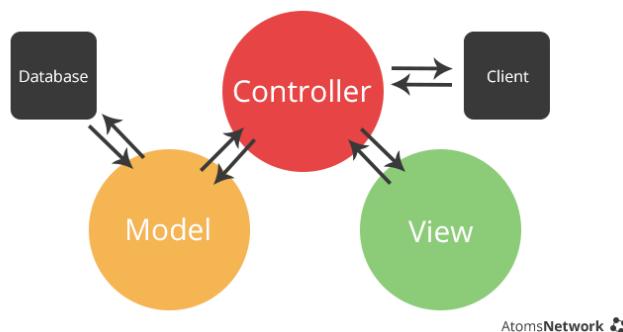
d. Tidak adanya tipe data pada *PHP*

PHP tidak memiliki tipe data. hal ini membuat kadang pada saat menggunakan bahasa pemrograman *PHP* muncul bug bug yang tidak diinginkan berkaitan dengan tidak adanya standar tipe data pada php. contohnya adalah data “1000” dan “1e3” jika dibandingkan akan memiliki tipe data yang sama karena secara implisit data tersebut dirubah menjadi *floating point*. namun kekurangan ini berkaitan sekali dengan pengalaman dari developer yang menggunakan bahasa pemrograman *PHP*. *developer* yang sudah ahli tentunya sudah paham betul bagaimana mengatasi permasalahan ini.

2.2.2.3 Hubungan *PHP* Dengan *Framework CodeIgniter*

Hubungan antara *PHP* dan *framework CodeIgniter* adalah bahasa pemrograman *PHP* merupakan bahasa pemrograman yang digunakan pada *framework CodeIgniter*.

2.2.3 *MVC (Model View Controller)*



Gambar 2.12 Ilustrasi MVC

Pada pembahasan sebelumnya sudah dijelasakan apa itu *MVC*. Pada sub bab ini akan dijelaskan lebih detail tentang *MVC*.

2.2.3.1 Konsep *MVC*

MVC (*Model-View-Controller*) adalah pola arsitektur yang memisahkan aplikasi dalam tiga komponen utama Logis: *Model*, *View* dan *Controller*. Masing - masing komponen ini dibangun untuk menangani aspek-aspek tertentu pembangunan aplikasi. *MVC* adalah salah satu kerangka pembangunan web standar industri paling sering digunakan untuk menciptakan proyek yang berukuran besar dan *extensible*.

2.2.3.2 Komponen *MVC*

1. ***Model***: Komponen *Model* yang sesuai dengan semua data yang terkait dengan penggunaan logika dalam berkerja. Ini dapat mewakili baik data yang ditransfer antara *View* dan *Controller* komponen atau logika bisnis lain data yang terkait. Sebagai contoh, sebuah objek pelanggan akan mengambil informasi pelanggan dari *database*, memanipulasi itu dan memperbarui data kembali ke *database* atau menggunakan untuk membuat data.
2. ***View***: Komponen *View* digunakan untuk *semua UI (User Interface)* pada logika aplikasi. Misalnya, tampilan pelanggan akan mencakup semua komponen *UI* seperti kotak teks, *dropdown*, dll yang digunakan pengguna untuk berinteraksi.
3. ***Controller***: *Controller* bertindak sebagai antar muka antara ***Model*** dan ***View*** komponen proses semua logika bisnis dan permintaan masuk, memanipulasi data menggunakan

komponen *Model* dan berinteraksi dengan *View* untuk membuat hasil akhir. Sebagai contoh, **controller** pelanggan akan menangani semua interaksi dan masukan dari *View* pelanggan dan update *database* menggunakan *Model* pelanggan. *Controller* sama akan digunakan untuk melihat data pelanggan.

2.2.3.3 *ASP.NET MVC*

ASP.NET mendukung pengembangan pada 3 model besar: Halaman *Web*, formulir *Web* dan *MVC* (*Model View Controller*). Kerangka *ASP.NET MVC* adalah *framework* yang ringan, sangat diuji presentasi yang terintegrasi dengan fitur **ASP.NET** yang ada, seperti halaman master, otentikasi, dll. Dalam *.NET*, kerangka kerja ini didefinisikan dalam Majelis *System.Web.Mvc*. Versi terbaru dari *Framework MVC* adalah 5.0. Kami menggunakan *Visual Studio* untuk membuat *ASP.NET MVC* aplikasi yang dapat ditambahkan sebagai template dalam *Visual Studio*.

2.2.3.4 Fitur *ASP.NET MVC*

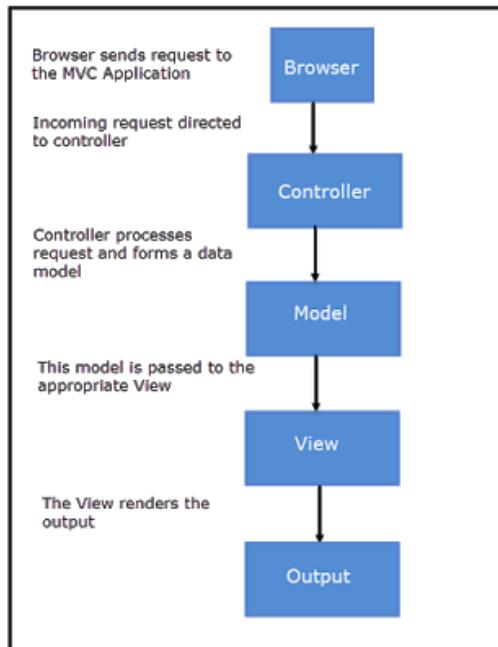
ASP.NET MVC menyediakan fitur berikut:

- a. Ideal untuk mengembangkan aplikasi dengan skala besar / kompleks tapi tetap ringan.
- b. Menyediakan kerangka *extensible* dan *pluggable* yang dapat dengan mudah diganti dan disesuaikan. Misalnya, jika Anda tidak ingin menggunakan *built-in Razor or ASPX View Engine*, maka Anda dapat menggunakan *view* mesin pihak ketiga atau bahkan menyesuaikan yang sudah ada.
- c. Memanfaatkan desain berbasis komponen aplikasi dengan logis membagi menjadi

komponen **Model**, **View** dan **Controller**. Hal ini memungkinkan para pengembang untuk mengelola kompleksitas proyek besar-besaran dan bekerja pada komponen individual.

- d. Struktur **MVC** meningkatkan pembangunan berbasis tes dan testability aplikasi karena semua komponen yang dapat dirancang dengan antar muka berbasis dan diuji menggunakan *mock* objek. Maka *Framework ASP.NET MVC* sangat ideal untuk proyek-proyek dengan tim besar pengembang *web*.
- e. Mendukung semua yang ada luas *ASP.NET* fungsionalitas seperti otorisasi dan otentikasi, Master halaman, Data *Binding*, kontrol pengguna, keanggotaan, *ASP.NET Routing*, dll.
- f. Tidak menggunakan konsep negara *View* (yang hadir dalam *ASP.NET*). Ini membantu dalam membangun aplikasi yang ringan dan memberikan kontrol penuh untuk para pengembang.

2.2.3.5 Diagram Alir MVC



Gambar 2.13 Diagram Alir MVC

Langkah-langkah aliran MVC

- a. *Browser* klien mengirimkan permintaan ke aplikasi *MVC*.
- b. *Global.ascx* menerima permintaan ini dan melakukan routing berdasarkan *URL* permintaan masuk menggunakan *RouteTable*, *RouteData*, *UrlRoutingModule* dan *MvcRouteHandler* objek
- c. Operasi ini *routing* panggilan *controller* sesuai dan mengeksekusinya menggunakan objek menggunakan metode *ControllerFactory* dan *MvcHandler* objek.
- d. *Controller* memproses data menggunakan *Model* dan memanggil metode yang tepat menggunakan *Controller Action Invoker* objek

- e. **Model** diproses kemudian dilewatkan ke **View** yang pada gilirannya membuat Keluaran terakhir.

2.2.3.6 Kegunaan *MVC*

Design pattern ini dikembangkan dengan tujuan untuk membuat sebuah program yang dapat dipergunakan secara berulang kali untuk hal yang serupa, dan dikembangkan dengan modul tambahan sehingga tidak terjadi proses pengulangan pengembangan dari nol. *Don't reinvent the wheel* – pepatah yang seringkali didengungkan di dunia pengembangan aplikasi, merupakan salah satu tujuan utama pemanfaatkan design pattern *MVC*.

Pada pemrograman *web* sebelumnya, *programmer* harus menghabiskan waktu yang sangat banyak untuk melakukan pengubahan fitur dalam aplikasi berbasis *web* atau *website* yang dikembangkannya. Seringkali pengubahan program tanpa disengaja mengubah juga bagian tampilan yang telah dibuat sebelumnya.

Dengan memanfaatkan design pattern ini, diharapkan *programmer* tidak lagi dipusingkan apabila *client* meminta *programmer* untuk mengubah tampilan dari program. Dalam hal ini *programmer* bisa memfokuskan perhatiannya pada bagian *View*.

Demikian pula apabila terjadi penambahan fitur pada aplikasi yang mengakibatkan pengubahan pada *logic* dari program serta perubahan pada basis data. *Database designer* dan *programmer* dapat bekerjama dalam mengubah Model maupun *Controller* tanpa harus terlalu bingung apa yang mereka kerjakan akan mempengaruhi tampilan.

Penggunaan *MVC* juga seringkali membuat implementasi aplikasi lebih sederhana dan jumlah baris program yang harus dibuat lebih minim. Fitur ini terutama yang menjadi dasar pengembangan berbagai *framework* yang telah disebutkan sebelumnya.

2.2.3.7 Framework Yang Menggunakan Konsep *MVC*

Untuk framework PHP, ada cukup banyak dan hampir semuanya menggunakan konsep *MVC*:

1. Laravel
2. Symfony
3. CakePHP
4. Zend
5. Codeigniter (versi 3 kebawah sudah tidak recommended untuk dipelajari)
6. Dll

Untuk framework Python di antaranya:

1. Django
2. Turbogears2
3. Watson-Framework
4. Dll

Untuk framework Nodejs di antaranya:

1. Express
2. Adonis
3. Sails.js
4. Total.js
5. Mean.js
6. Mojito
7. Dll

2.2.3.8 Contoh Kasus

Ketika anda memasuki perpustakaan kemungkinan anda diminta untuk mengisi buku tamu. Kebetulan buku tamu di perpustakaan yang anda kunjungi menggunakan aplikasi web. Jadi di sana sudah terdapat satu komputer, di mana setiap pengunjung perpustakaan harus melewati dan mengisi datanya melalui komputer tersebut.

1. Ketika anda melihat layar komputer, di sana ditampilkan form untuk mengisi data diri pengunjung. Kita bisa simpulkan bahwa di sini, browser (sebagai client) meminta aksi kepada server (yang ditangani oleh controller) untuk menampilkan halaman form input data. Lalu controller memutuskan dan mengerti bahwasanya ia hanya perlu menampilkan view. Maka controller memanggil dan mengembalikan view (atau halaman) yang diminta.
2. Anda kemudian mengisi data. Lalu menekan tombol submit. Di sini browser (sebagai client) mengirimkan data anda kepada server. Data itu ditangkap oleh controller dan controller tahu apa yang harus dia lakukan. Yaitu memanggil model dan memberi data tersebut untuk kemudian disimpan (oleh model) ke dalam database. Setelah proses penyimpanan selesai, controller memanggil dan mengembalikan view kepada user sebagai informasi bahwa data telah masuk.
3. Jika ada pengunjung baru, maka kembali ke step 1.

2.2.4 Fungsi *CodeIgniter*

1. Mempercepat dan mempermudah kita dalam pembuatan website.

2. Menghasilkan struktur pemrograman yang sangat rapi, baik dari segi kode maupun struktur file phpnya.
3. Memberikan standar coding sehingga memudahkan kita atau orang lain untuk mempelajari kembali system aplikasi yang dibangun.

2.2.5 Kelebihan dan Kekurangan *CodeIgniter*

a. Kelebihan *CodeIgniter*

1. Berukuran sangat kecil. File download nya hanya sekitar 2MB, itupun sudah includedokumentasinya yang sangat lengkap.
2. Dokumentasi yang bagus. Saat anda mendownloadnya, telah disertakan dengan dokumentasi yang berisi pengantar, tutorial, bagaimana panduan penggunaan, serta referensi dokumentasi untuk komponen-komponennya.
3. Kompatibilitas dengan Hosting. CodeIgniter mampu berjalan dengan baik pada hampirsemua platfom hosting. CodeIgniter juga mendukung database-database paling umum, termasuk MySQL.
4. Tidak ada aturan coding yang ketat. Terserah anda jika anda hanya ingin menggunakan Controller, tanpa View, atau tidak menggunakan Model, atau tidak salah satu keduanya. Namun dengan menggunakan ketiga komponennya adalah pilihan lebih bijak.
5. Kinerja yang baik. Codeigniter sangat cepat bahkan mungkin bisa dibilang merupakan framework yang paling cepat yang ada saat ini.

6. Sangat mudah diintegrasikan. CodeIgniter sangat mengerti tentang pengembangan berbagai library saat ini. Karenanya CodeIgniter memberikan kemudahan untuk diintegrasikan dengan library-library yang tersedia saat ini.
7. Sedikit Konfigurasi. Konfigurasi CodeIgniter terletak di folder application/config. CodeIgniter tidak membutuhkan konfigurasi yang rumit, bahkan untuk mencoba menjalankannya, tanpa melakukan konfigurasi sedikitpun ia sudah bisa berjalan.
8. Mudah dipelajari. Disamping dokumentasi yang lengkap, ia juga memiliki berbagai forum diskusi.

b. Kekurangan *CodeIgniter*

1. CodeIgniter tidak ditujukan untuk pembuatan web dengan skala besar.
2. Library yang sangat terbatas. Hal ini dikarenakan sangat sulit mencari plugin tambahan yang terverifikasi secara resmi, karena pada situsnya CodeIgniter tidak menyediakan plugin-plugin tambahan untuk mendukung pengembangan aplikasi dengan CI.
3. Belum adanya editor khusus CodeIgniter, sehingga dalam melakukan create project dan modul-modulnya harus berpindah-pindah folder.

2.3 E-Mail



Gambar 2.14 Ilustrasi E-Mail

2.3.1 Apa itu E-Mail?

E-mail atau surat elektronik adalah suatu alat atau sarana untuk mengirim dan menerima surat atau pesan dengan format dalam bentuk digital melalui jalur jaringan komputer dan internet.

2.3.2 Elemen-Elemen E-Mail

Menurut David Alex Lamb (1999), dalam proses pengiriman dan penerimaan *email* terdapat beberapa elemen yang sangat berpengaruh terhadap proses tersebut. Elemen-elemen yang dimaksud adalah :

1. *Sender*, merupakan orang yang menyusun dan mengirimkan *email*.
2. *Mail agent*, merupakan perangkat lunak yang digunakan oleh pengirim untuk melakukan penyusunan *email*.
3. *Message*, merupakan representasi komputer dari apa yang ingin disampaikan oleh pengirim.
4. *Email transport subsystem*, merupakan sebuah sistem yang menangani transportasi *email*.
5. *Receiver*, merupakan tujuan atau orang yang menerima *email*.

6. *Email agent software*, merupakan sebuah perangkat lunak yang digunakan untuk membaca *email*.
7. *Email address*, merupakan sekumpulan karakter yang digunakan untuk mengenali pengirim dan penerima.

2.3.3 Perbandingan *E-Mail* Dengan Surat Biasa

Pada dasarnya sistem pengiriman dan penerimaan *e-mail* sama seperti pada sistem penerimaan dan pengiriman surat biasa. Berikut merupakan perbandingan antara *e-mail* dan surat biasa.

Tabel 2.1 Perbandingan E-mail dengan surat biasa

Pembanding	<i>E-mail</i>	Surat Biasa
<i>Mail Agent</i>	<i>Outlook Express</i>	Pena/Mesin Tik
Media Pesan	<i>Text, HTML</i>	Kertas, Amplop
<i>Subsystem</i>	<i>SMTP, POP3, IMAP</i>	POS, <i>DHL</i> , dan jasa kurir lainnya
Alamat	<i>E-mail</i>	Alamat Rumah

Dalam sistem *e-mail* terdapat dua buah *subsystem e-mail*.

1. MUA (*Mail User Agent*) merupakan sebuah program di komputer lokal yang mendukung penggunaan *command-based* (berbasis perintah), *menu based* (berbasiskan menu) atau grafikal. MUA digunakan untuk membaca dan mengirimkan *e-mail*. Contoh aplikasi dari MUA antara lain *Outlook Express*, *KMail*, *Eudora*, *Pine*.
2. Subsistem kedua adalah MTA (*Mail Transport Agent*) merupakan sebuah *daemon system* yang berjalan di belakang (*background*)

dalam proses pemindahan *e-mail* dimana fungsi MTA adalah sebagai pengatur transportasi *e-mail* dari pengirim ke penerima.

2.3.4 Format *E-mail*

Sebuah pesan merupakan sebuah kumpulan karakter yang dipisahkan ke dalam beberapa baris karakter. Baris karakter yang digunakan dibatasi oleh dua buah karakter berupa *Carriage Return* (CR) dengan nilai ASCII 13, yang kemudian diikuti langsung oleh *Line Feed* (LF) dengan nilai karakter ASCII 10 (RFC 2822).

E-mail terdiri dari dua bagian utama, *header* dan *body*. Bagian *header* minimal terdiri dari tiga isian utama berupa “*date*”, “*from* : “, dan “*to*”, dimana setiap *field* pada *header* memiliki aturan penulisan sendiri.

Beberapa *field* yang umum digunakan pada bagian *header*:

Tabel 2.2 Field header pada e-mail

<i>Header Field</i>	Berisi
<i>To:</i>	Satu atau lebih alamat <i>email</i> penerima primer
<i>CC:</i>	Satu atau lebih alamat <i>email</i> penerima sekunder
<i>BCC:</i>	Satu atau lebih alamat <i>email</i> penerima <i>blind carbon copy</i>
<i>From:</i>	Orang yang membuat <i>email</i>
<i>Subject:</i>	Judul <i>email</i>
<i>Sender:</i>	Alamat <i>email</i> pengirim
<i>Received:</i>	Isian yang ditambahkan setiap melalui MTA
<i>Return Path:</i>	Jalur kembali mail ke pengirim

2.3.5 Tahapan Proses Pengiriman *E-mail*

1. Pengiriman pesan

Pada proses ini, pengirim menyusun pesan dan mengirimkannya dengan *email agent*, yang akan memberikan perintah kepada sistem *transport email* untuk mengantarkan pesan tersebut ke tujuan.

a. *Composing*

Penyusunan *e-mail* mirip dengan penyusunan naskah, dimana *e-mail agent* pada umumnya memiliki beberapa editor-editor teks.

Yang membedakan adalah format *e-mail* yang terdiri dari dua jenis, *header* dan *body*. Mengacu pada RFC 822, *header* merupakan bagian *e-mail* yang terdiri dari alamat, judul dan tembusan. Sedangkan *body* merupakan bagian dari *e-mail* yang memuat pesan yang akan disampaikan oleh pembuat *e-mail* berupa teks murni atau disertakan *file* dengan format tertentu yang nantinya dikodekan sebagai teks yang disebut dengan MIME (*Multipurpose Internet Mail Extensions*). *E-mail agent* biasanya menyediakan ruang khusus sebagai tempat menaruh konsep atau rancangan *e-mail* yang sudah dibuat sehingga pengguna bisa melakukan *editing* kembali.

b. Pengantrian dan pengiriman

Setelah *e-mail* disusun dan kemudian pengguna memutuskan untuk mengirimkan pesan tersebut dengan menggunakan perintah tertentu, maka *e-mail agent* akan memeriksa alamat tujuan (sebagai contoh *header field to:*) sesuai dengan aturan penulisan alamat *e-mail* tujuan atau belum. Jika penulisan salah maka *e-mail agent* akan memberikan pemberitahuan kepada pengguna untuk memeriksa ulang dan memperbaiki. Sebaliknya jika penulisan alamat *e-mail* tujuan sudah benar, maka *e-mail* akan dikirimkan.

Dimana *e-mail agent* akan memanggil *e-mail transport subsystem* untuk mengirimkan *e-mail*.

Beberapa pengguna koneksi internet *dial up* menyusun *e-mail* pada saat *off line* dan baru akan mengirimkannya pada saat *online*.

Kasus lain terjadi ketika suatu *e-mail* tidak bisa dikirimkan karena ada *e-mail* lain yang memiliki prioritas lebih tinggi. *Mail transport* dan *mail agent* memiliki tempat antrian khusus yang digunakan untuk menampung *email* yang akan dikirimkan, jika kondisi memungkinkan maka *e-mail-e-mail* yang ditampung tersebut akan dikirim ke komputer tujuan.

2. Pemindahan

a. Pengalamanan *e-mail*

Satu atau lebih alamat tujuan oleh *mail transport system* digunakan untuk mengetahui kemana *e-mail* akan dikirimkan. Alamat *e-mail* yang digunakan berupa sekumpulan teks yang mengidentifikasi keberadaan kotak *e-mail* (*mailbox*) pada IP atau *domain* tertentu.

Layanan *transport* mengirimkan kumpulan teks *site* pada *Domain Name System* (DNS) yang akan mengubahnya ke dalam alamat IP kemudian menghubungkan ke alamat IP yang diberikan dan meminta komputer tujuan untuk menerima dan mengirimkannya pada *mailbox* yang dituju.

b. *E-mail server*

Ketika *email* dikirimkan dan sebelum sampai kepada *mailbox* yang dituju, terlebih dahulu isi pesan *email* tersebut diproses pada *e-mail server*. Ada tiga hal yang dilakukan *e-mail server* sebagai respon dari permintaan tersebut, berupa :

- i. Menerima pesan dan menyampaikannya dalam *mailbox* tujuan,

- ii. Menerima pesan ke alamat lain yang telah ditetapkan oleh pemilik *mailbox*,
- iii. Menolak pesan dan memberikan pemberitahuan bahwa pesan tidak terkirim yang diakibatkan karena *mailbox* tujuan tidak tersedia, *mailbox* tujuan dalam kondisi penuh atau karena terdapat kerusakan pada *server*.

Dalam penggunaannya terdapat dua buah aplikasi *mail server*, kedua aplikasi tersebut berupa :

- i. SMTP

Protokol yang digunakan untuk mengelola lalu lintas *email* keluar masuk suatu jaringan.

- ii. POP

Protokol yang digunakan untuk mengambil *email* dari tempat penampungan *e-mail* pada *email server*.

3. Penerimaan *e-mail*

Suatu saat *mail agent* melakukan pengecekan secara otomatis atau berdasarkan permintaan pengguna, apakah ada *e-mail* masuk ke dalam *mailbox* pada *e-mail server*. Jika terdapat *email* masuk maka kemudian *mail agent* tersebut akan menyimpan *e-mail* tersebut dalam *database mail agent* tersebut.

2.3.6 Layanan *E-mail*

Terdapat lima layanan dasar yang didukung oleh sistem *e-mail* menurut Tanenbaum, kelima dasar tersebut adalah :

1. Komposisi

Berhubungan dengan proses pembuatan pesan dan/atau balasan *email*, pengolah teks digunakan untuk badan *email* dan sistem

membantu pengalamatan dan beberapa isian pada bagian kepala (*header*) *email*.

2. Pengiriman

Ketika seorang pengirim melakukan proses pengiriman sebuah pesan kepada penerima maka ketika itu juga diperlukan sebuah hubungan dari pengirim kepada penerima, atau media lain yang berada diantaranya. Setelah koneksi dilakukan, pemutusan koneksi dilakukan jika *email* telah dikirimkan. Pengiriman ini merupakan proses *background* pada sistem.

3. Pelaporan

Memberikan laporan kepada pengirim terhadap kejadian yang dialami oleh pesan. Kondisi yang ditemukan berupa laporan keberhasilan pengiriman pesan selain itu bisa juga pelaporan berupa penolakan atau hilangnya pesan yang dikirim.

4. Tampilan

Format tampilan pada layar komputer penerima sesuai dengan format yang telah dibuat oleh pengirim. Meskipun pada kenyataannya terdapat perbedaan format MUA antara penerima dan pengirim.

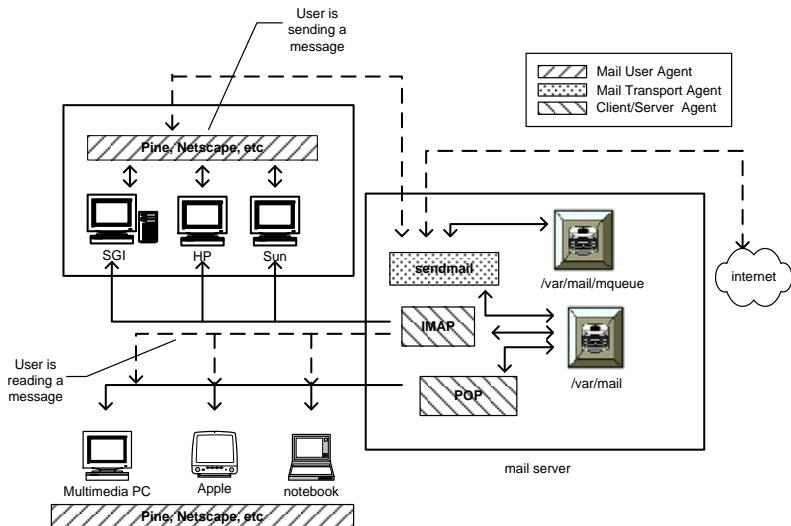
5. Penempatan

Berhubungan dengan apa yang akan dilakukan penerima setelah menerima *e-mail*.

2.3.7 Komponen *E-mail*

Sebuah *email* secara umum memiliki arsitektur seperti yang terlihat pada gambar 2.6, dengan sejumlah komponen penyusun. Pesan yang dikirimkan akan tersimpan di dalam *email server* yang dikirimkan melalui *mail user agent*. *Mail user agent* mengambil dan

mengirimkan pesan tersebut ke dan dari tempat penyimpanan melalui protokol internet yang dapat diandalkan.



Gambar 2.15 Arsitektur E-mail Client/Server

Sebuah *email* terdiri dari beberapa komponen penyusun. Komponen-komponen tersebut adalah :

1. *Mail User Agent*

Merupakan program yang digunakan untuk membuat dan membaca *email*. MUA bisa disebut juga sebagai *email reader* yang dapat menerima bermacam - macam perintah untuk pembuatan, penerimaan dan penjawaban pesan. Selain itu MUA juga dapat digunakan untuk memanipulasi *mailbox email* pengguna.

Beberapa MUA mengizinkan pemakaian format *Multipurpose Internet Mail Extension* (MIME) yang dapat digunakan untuk melampirkan *file* ke dalam suatu pesan. Dengan MIME ini, pesan yang dikirimkan dapat melampirkan *file-file* lain. *MailX*, *pine* dan *netscape* merupakan beberapa contoh MUA.

2. *Mail Transport Agent*

Pada saat proses pengiriman *email*, *email* tersebut diberikan ke MTA untuk dilakukan proses yang lebih lanjut.

Fungsi dari MTA adalah sebagai berikut :

- a. MTA menggunakan alamat tujuan untuk menentukan bagaimana pesan tersebut harus dikirimkan.
- b. MTA dapat menggunakan *aliases*/daftar distribusi untuk mengirimkan salinan dari sebuah pesan ke berbagai tujuan.
- c. MTA menerima dan memproses *email* yang masuk dari mesin lain dalam jaringan.

3. *Mail Channel* dan *Delivery Agent*

Mail channel memiliki dua komponen utama yaitu :

- a. Tabel yang menentukan saluran yang akan digunakan untuk mengirimkan pesan *email*.
- b. *Delivery agent* yang akan melakukan pengiriman pesan bagi saluran yang telah ditentukan.

MTA akan mengenali saluran pengiriman pertama yang sesuai dengan alamat pesan pada tabel dan menggunakan agen pengiriman yang sesuai untuk mengirimkan pesan. Alamat yang tidak sesuai dengan saluran yang ada akan dianggap sebagai alamat yang tidak dapat dikirimkan dan akan dikembalikan ke pengirim. Jenis-jenis saluran yang digunakan :

- a. *Local channel* yang menangani pengiriman *email* ke pengguna lokal.
- b. *SMTP channel* akan mengirimkan *email* melalui jaringan TCP/IP dengan menggunakan DNS dan SMTP.

- c. *Badhost channel* yang secara default akan mengantrikan pesan-pesan yang sementara waktu tidak dapat dikirimkan karena nama *server* tidak siap.

Pesan *email* akan *di-route* dari pengirim ke penerima oleh MTA pada *host email* yang berbeda. Komunikasi diantara *host* yang berbeda ditangani oleh program yang disebut *mailer*.

Beberapa *mailer* yang biasa digunakan adalah :

- a. *Local* : pengiriman *local*
- b. *Ether* : pengiriman *Ethernet*
- c. *Ddn* : pengiriman *internet*
- d. *Uucp* : pengiriman *uucp*
- e. *Error* : *mailer* yang digunakan untuk menghasilkan pesan *error* ke pengguna.

2.3.8 Multipurpose Internet Mail Extension

Pada awalnya *email* ditulis dalam bahasa inggris dengan menggunakan format ASCII, namun seiring dengan perkembangan zaman kebutuhan akan *email* sudah merambah tidak hanya oleh pengguna yang berbasiskan ASCII saja. Untuk itu diperlukan sebuah cara yang dapat merepresentasikan suatu huruf lain dalam sebuah *email*. RFC 2045 mendefinisikan ulang format pesan MIME ditujukan untuk menangani:

1. Pesan yang digunakan bukan dalam bentuk US –ASCII.
2. Sekumpulan bentuk pesan yang bukan teks.
3. Pesan dengan sifat *Multi-part*.
4. Informasi *header mail* yang digunakan bukan dalam karakter US-ASCII.

Selain itu permasalahan pengiriman *email* yang bisa diatasi dengan MIME berupa :

1. Pesan dalam aksara aksentuasi (contoh : Prancis)
2. Pesan dalam aksara *non latin* (contoh : Rusia)
3. Pesan dalam bentuk aksara tanpa alfabet (contoh : Jepang)
4. Pesan dalam bentuk *non-text* (contoh : gambar dan video)

Tabel 2.3 Header tambahan MIME

Header	Fungsi
<i>MIME – Version :</i>	Identifikasi versi MIME
<i>Content-description :</i>	Memberitahukan apa yang ada dalam pesan
<i>Content-ID :</i>	Pengenal khusus / unik
<i>Content-Transfer-Encoding :</i>	Membungkus pesan untuk pengiriman
<i>Content-Type :</i>	Memberitahukan type pesan

MIME didefinisikan dengan lima *header* baru. *Header MIME text* digunakan untuk memberitahukan *mail user agent* apakah pesan berupa *plaintext* atau bukan. Jika pesan tidak memiliki “*MIME-Version*:

“pesan tersebut diasumsikan dalam format *plaintext* dan beraksara inggris. *Header “Content-Description:*” berisi pesan ASCII, berfungsi untuk memberitahukan apa yang ada dalam pesan kepada si penerima *email* tersebut, sehingga penerima pesan bisa memutuskan apakah pesan tersebut akan dibaca, dikodekan atau dibuang.

Header “Content-ID:” merupakan identitas dari MIME yang bersifat unik. *Header “Content-Transfer-Encoding:*” digunakan untuk

menentukan bagaimana pesan dikemas untuk pengirimannya melalui suatu jaringan. Untuk data biner pada umumnya menggunakan *base64* dan *quoted printable encoding*. Jika pesan lebih banyak berupa kode ASCII daripada *non ASCII* maka yang digunakan adalah *quoted printable encoding*, karena lebih efisien dibandingkan menggunakan *base64*. Sedangkan “*Content-Type*” mengidentifikasi sifat atau karakter dari pesan.

Tabel 2.4 Konten Type

Tipe	Subtipe	Deskripsi
<i>Text</i>	<i>Plain</i>	Teks murni atau <i>unformatted text</i> .
	<i>Richtext</i>	Teks dengan perintah pemformatan sederhana.
<i>Image</i>	<i>GIF</i>	Gambar dalam bentuk GIF.
	<i>JPEG</i>	Gambar dalam bentuk JPEG.
<i>Audio</i>	<i>Basic</i>	<i>Audible Sound</i> .
<i>Video</i>	<i>MPEG</i>	Film dan format MPEG.
<i>Application</i>	<i>Octet-Stream</i>	Rangkaian byte yang belum diterjemahkan (uninterpreted).
	<i>Postscript</i>	Dokumen <i>printable</i> dalam format Postscript.
<i>Message</i>	<i>RFC822</i>	Pesan dalam bentuk MIME RFC 822.
	<i>Partial</i>	Pesan dipecah untuk kepentingan pengiriman.
	<i>External Body</i>	Pesan diperoleh dari jaringan internet.
<i>Multiport</i>	<i>Mixed</i>	Mengijinkan setiap bagian pesan memiliki tipe berbeda.

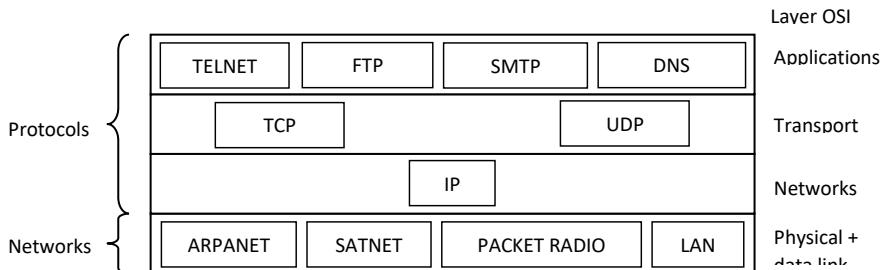
	<i>Alternative</i>	Pesan yang sama dalam format yang berbeda.
	<i>Parallel</i>	Semua bagian pesan ditampilkan secara simultan.
	<i>Digest</i>	Setiap bagian <i>email</i> dalam bentuk pesan RFC 822 komplet.

2.3.9 Protokol *E-mail*

1. *Simple Mail Transport Protocol*

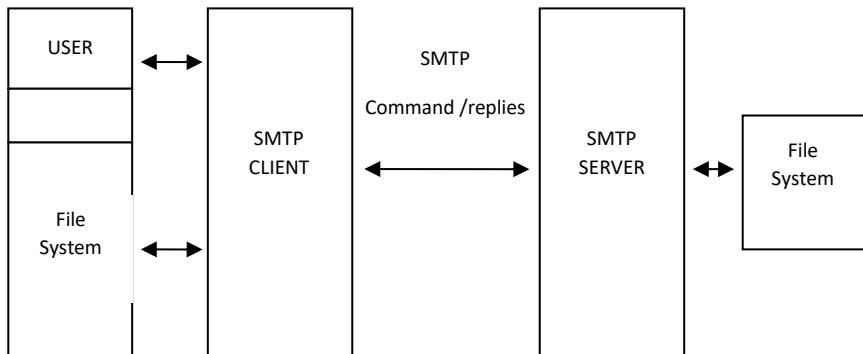
Seperti yang sudah dibahas sebelumnya SMTP merupakan protokol yang digunakan untuk mengelola lalu lintas *email* keluar masuk suatu jaringan.

SMTP berada pada *layer* aplikasi dengan *port* 25. Fungsi utama dari SMTP adalah menyampaikan pesan dari komputer pengirim ke komputer penerima, baik dalam jaringan yang sama (LAN atau WAN) maupun ke komputer penerima dalam suatu jaringan yang berbeda.



Gambar 2.16 Layer OSI

Model SMTP yang dijelaskan pada RFC 2821 berupa :



Gambar 2.17 Model SMTP

Jika akan mengirimkan suatu *email*, maka *SMTP Client* akan membuka kanal dua arah ke *SMTP Server*. Dalam hal ini *SMTP Server* bisa merupakan tujuan akhir, namun kadang bisa juga menjadi perantara antara komputer penerima dengan komputer pengirim atau berupa gerbang yang menghubungkan komunikasi SMTP dengan protokol lain.

Koneksi *SMTP Client-Server* diawali dengan proses inisialisasi, *SMTP Server* akan memberikan status bisa digunakan atau tidak. Jika tidak bisa digunakan maka koneksi diputus dan jika statusnya bisa digunakan *SMTP client* bisa memulai pengiriman kumpulan perintah yang diperlukan seperti menentukan alamat pengirim, alamat tujuan, serta pesan yang akan disampaikan. Setelah pesan dikirimkan oleh *SMTP server*, *SMTP client* bisa meminta koneksi diputus atau dimulai untuk pengiriman *email* lainnya.

2. *Internet Message Access Protocol* dan *Post Office Protocol*

Protokol IMAP dan POP digunakan untuk dapat menjembatani *email user* dan *server*. Hal ini dikarenakan kondisi *email user* yang digunakan tidak secara terus menerus terkoneksi sehingga *email* yang masuk akan ditampung pada *email server*. Kemudian jika akan

dibaca, *email* tersebut didownload oleh pengguna setelah terkoneksi dan berinteraksi dengan *email server*.

a. *Post Office Protocol*

POP merupakan protokol yang digunakan untuk mengambil pesan dari *mailbox* pada komputer *server* dan menyimpannya pada komputer lokal pengguna POP3. *Server* menggunakan *port* 110 pada TCP/IP. Jika ada *client* yang akan menggunakan layanan *server*, maka koneksi antara keduanya dilangsungkan. Setelah terkoneksi, *server* POP3 akan memberikan sebuah pesan sambutan yang kemudian dilanjutkan pada tahap berikutnya yaitu tahapan otorisasi, dimana *client* harus mengidentifikasi dirinya ke *server* POP3 dengan mengirimkan *user id* dan *password*.

Jika otorisasi berhasil dan sesuai dengan data yang tersedia di *server*, maka *server* akan mengambil data yang dibutuhkan dalam koneksi tersebut dan dilanjutkan dengan tahapan transaksi.

Pada tahapan transaksi, pengguna bisa menggunakan beberapa perintah untuk berinteraksi dengan *server*, semisal menampilkan daftar *email* yang tersedia dalam *mailbox*. Semua pesan yang dikirimkan dalam koneksi POP3 berupa kode ASCII dan format pesan *email* yang dikirimkan diasumsikan sesuai dengan standar pada RFC 822. Karena semua pesan tidak terenkripsi jika dilakukan penyadapan, maka pesan yang dikomunikasikan selama koneksi dapat dibaca langsung. Solusi untuk permasalahan ini dapat dibangun dengan koneksi SSH, menggunakan otentikasi/enkripsi (PGP atau PEM) pada pesan *email*.

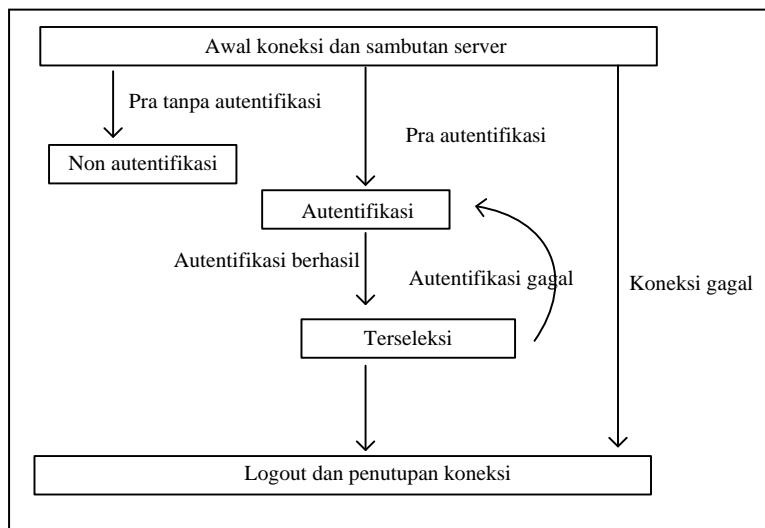
b. *Internet Message Access Protocol*

IMAP merupakan sebuah protokol yang dirancang agar *user* dapat mengakses *email* pada *mailbox* serta dapat berinteraksi dengan *mail*

server. Port yang digunakan oleh protokol ini dalam TCP/IP adalah port nomor 143.

Sesi koneksi IMAP terdiri dari koneksi *client – server*, pesan sambutan dan interaksi *client – server*. Interaksi *client – server* terbagi menjadi perintah *client*, *data server* dan tanggapan penyelesaian perintah *server*. Koneksi IMAP terbagi menjadi empat kondisi, yaitu :

- Non-autentifikasi, dilakukan agar pengguna mendapatkan hak untuk menjalankan perintah dalam koneksi. Jika tidak bisa dilakukan, pengguna hanya bisa memberikan perintah yang bisa dijalankan pada semua keadaan berupa *Capability*, *Noop* dan *Logout*.
- Terautentifikasi, pada jenis ini pengguna diberikan hak akses dan pengiriman perintah. Untuk dapat melakukan ini *user* harus memiliki *mailbox* untuk diakses sesuai dengan wewenangnya sebelum perintah-perintah yang dapat mempengaruhi pesan dikirimkan ke *server*.
- Terseleksi, keadaan ini dimulai setelah *mailbox* yang ditentukan dapat diakses oleh *user*.
- *Logout* (keluar), kondisi ini akan memutuskan koneksi. Kondisi ini terjadi karena pemutusan koneksi yang disengaja oleh *user*, selain itu bisa juga diakibatkan karena sesuatu hal semisal tidak terjadinya interaksi dalam waktu tertentu.



Gambar 2.18 Koneksi Client – Server IMAP

2.4 Penjelasan *Tools* Yang Digunakan

2.4.1 *Visual Studio Code*



Gambar 2.19 Visual Studio Code

1. Pengenalan *Visual Studio Code*

Visual Studio Code (VS Code) adalah sebuah teks editor ringan dan handal yang dibuat oleh Microsoft untuk sistem operasi multiplatform, artinya tersedia juga untuk versi Linux, Mac, dan Windows. Teks editor ini secara langsung mendukung bahasa pemrograman JavaScript, Typescript, dan Node.js, serta bahasa pemrograman lainnya dengan bantuan plugin yang dapat dipasang via marketplace Visual Studio Code (seperti C++, C#, Python, Go, Java, dst).

Banyak sekali fitur-fitur yang disediakan oleh *Visual Studio Code*, diantaranya *Intellisense*, *Git Integration*, *Debugging*, dan fitur

ekstensi yang menambah kemampuan teks editor. Fitur-fitur tersebut akan terus bertambah seiring dengan bertambahnya versi *Visual Studio Code*. Pembaruan versi *Visual Studio Code* ini juga dilakukan berkala setiap bulan, dan inilah yang membedakan *VS Code* dengan teks editor-teks editor yang lain.

2. Fitur-Fitur *Visual Studio Code*

- a. **Cross platform** – tersedia di macOS, Linux dan Windows artinya Anda dapat bekerja pada sistem operasi manapun tanpa khawatir belajar coding tools yang sama untuk sistem yang berbeda-beda.
- b. **Lightweight** – tak perlu menunggu lama untuk memulai. Anda mengontrol sepenuhnya bahasa, tema, debugger, commands dan lain-lainnya sesuai keinginan. Ini dapat dilakukan melalui extentions untuk bahasa populer seperti python, node.js, java dan lain-lainnya di Visual Studio Code Marketplace.
- c. **Powerful editor** – memfungsikan fitur untuk source code editing yang sangat produktif, seperti membuat code snippets, IntelliSense, auto correct, dan formatting.
- d. **Code Debugging** – salah satu fitur terkeren yang ditawarkan Visual Studio Code adalah membantu Anda melakukan *debug* pada kode dengan cara mengawasi kode, variabel, call stack dan expression yang mana saja.
- e. **Source control** – Visual Studio Code memiliki integrated source control termasuk Git support in-the-box dan penyedia source code control lainnya di pasaran. Ini meningkatkan siklus rilis proyek Anda secara signifikan.

- f. **Integrated terminal** – Tiada lagi *multiple windows* dan alt-tabs. Anda dapat melakukan command-line task sekejap dan membuat banyak terminal di dalam editor.

3. *Plugin Visual Studio Code*

- a. *Faker*

Dengan memasang plugin ini kita bisa membuat *fake data* untuk alamat, nama perusahaan, tanggal, nama orang, nomor telepon, atau sekedar lorem ipsum. Daripada memasukkan "adsfasdfksdfj" lebih baik pasang Extension ini dan buat data yang lebih "nyata".

- b. *Prettier*

Pernah kan *copy paste* kode baru yang ternyata tidak terindentasi dengan sempurna? Sering kita atur indentasi kode manual dengan tombol TAB atau SHIFT+TAB. Prettier akan memformat kode kita secara otomatis sehingga kita tidak perlu membuat waktu yang untuk memformat dokumen. Tak hanya mengatur indentasi, ia juga bisa mengatur urutan import, memaksa penggunaan " untuk menggantikan ', mengatur spasi, penulisan parameter, dsb.

- c. *Color Info*

Plugin ini memberikan informasi seputar kode warna yang kita tulis di CSS. Dengan mengarahkan kursor ke kode hexa warna tertentu, kita akan diberikan preview warnanya beserta nilai-nilainya di berbagai format lain seperti rgb, hsl, cmyk, juga nilai alpha-nya.

- d. *TODO Highlight*

Plugin ini saat diaktifkan akan menyeleksi komentar TODO di kode kita sehingga bagian-bagian mana yang harus dikerjakan berikutnya bisa terlihat dengan jelas. Saat dipasang ia akan bisa mendeteksi keyword TODO dan FIXME tapi kita juga bisa menambah keyword sendiri.

e. *Minify*

Extension ini berguna untuk melakukan *minifying* kode JavaScript, CSS, maupun HTML dengan memanfaatkan [uglify-js](#), [clean-css](#), dan [html-minifier](#).

f. *Change Case*

Change Case membantu kita dalam memodifikasi strings, menjadi camelCase, kebab-case, snake_case, CONST_CASE, dll.

g. *CSS Peek*

Extension ini membuat kita bisa langsung mengakses definisi kelas CSS dari file HTML.

h. *Open In-Browser*

VSCode tidak bisa memiliki antarmuka yang langsung membawa kita untuk membuat file HTML ke browser default. Extension ini akan menambahkan menu *Open with Default Browser* saat dipasang yang akan langsung memuat file HTML tertentu ke Firefox, Chrome, atau IE.

i. *Quokka*

Quokka merupakan Extension untuk membantu melakukan live debugging. Saat menulis kode, Quokka akan langsung memberikan umpan balik berupa hasil dari kode yang dieksekusi.

j. *Git Lens*

Git Lens akan menunjukkan siapa saja anggota tim yang menuliskan bagian-bagian kode yang sedang kita buka. Dengan plugin ini kita bisa tahu siapa menulis kode yang mana.

2.4.2 XAMPP



Gambar 2.20 XAMPP

1. Pengenalan XAMPP

XAMPP adalah perangkat lunak bebas, yang mendukung banyak sistem operasi, merupakan kompilasi dari beberapa program. XAMPP merupakan *tool* yang menyediakan paket perangkat lunak ke dalam satu buah paket. Dengan menginstall XAMPP maka tidak perlu lagi melakukan instalasi dan konfigurasi *web server Apache*, *PHP* dan *MySQL* secara manual. XAMPP akan menginstalasi dan mengkonfigurasikannya secara otomatis untuk anda atau auto konfigurasi.

XAMPP merupakan salah satu paket installasi *Apache*, *PHP* dan *MySQL* instan yang dapat kita gunakan untuk membantu proses installasi ketiga produk tersebut. Selain paket installasi instant XAMPP versi 1.6.4 juga memberikan fasilitas pilihan penggunaan *PHP4* atau *PHP5*. Untuk berpindah versi *PHP* yang ingin digunakan juga sangat mudah dilakukan dengan menggunakan

bantuan *PHP-Switch* yang telah disertakan oleh *XAMPP*, dan yang terpenting *XAMPP* bersifat *free* atau gratis untuk digunakan.

XAMPP merupakan *tool* yang menyediakan paket perangkat lunak ke dalam satu buah paket. Dengan menginstall *XAMPP* maka tidak perlu lagi melakukan instalasi dan konfigurasi *web server Apache*, *PHP* dan *MySQL* secara manual. *XAMPP* akan menginstalasi dan mengkonfigurasikannya secara otomatis. Merupakan *web server* yang mudah digunakan yang dapat melayani tampilan halaman web yang dinamis. Untuk mendapatkanya dapat mendownload langsung dari web resminya.

2. Sejarah *XAMPP*

XAMPP merupakan pengembangan dari *LAMP* (*Linux Apache, MySQL, PHP and PERL*), *XAMPP* ini merupakan *project non-profit* yang dikembangkan oleh *Apache Friends* yang didirikan Kai ‘Oswalad’ Seidler dan Kay Vogelgesang pada tahun 2002, project mereka ini bertujuan mempromosikan penggunaan *Apache web server*.

3. Fungsi *XAMPP*

Fungsi *XAMPP* sendiri adalah sebagai *server* yang berdiri sendiri (*localhost*), yang terdiri beberapa program antara lain : *Apache HTTP Server*, *MySQL database*, dan penerjemah bahasa yang ditulis dengan bahasa pemrograman *PHP* dan *Perl*. Nama *XAMPP* sendiri merupakan singkatan dari *X* (empat sistem operasi apapun), *Apache*, *MySQL*, *PHP* dan *Perl*.

Program ini tersedia dalam *GNU (General Public License)* dan bebas, merupakan *web server* yang mudah untuk digunakan yang dapat menampilkan halaman *web* yang dinamis. Untuk mendapatkanya *XAMPP* anda dapat mendownload langsung dari *web* resminya. Dan berikut beberapa definisi program lainnya yang terdapat dalam *XAMPP*.

4. Fitur *XAMPP*

a. *Apache*

Apache adalah perangkat lunak sumber terbuka yang menjadi alternatif dari *server web Netscape*. *Server HTTP Apache* atau *Server Web/WWW Apache* merupakan *server web* yang dapat dijalankan di banyak sistem operasi yang berguna untuk melayani dan memfungsikan situs *web*. *Apache* dikembangkan oleh komunitas terbuka yang terdiri dari pengembang-pengembang dibawah naungan *Apache Software Foundation*.

b. *MySQL*

MySQL adalah singkatan “*My Structured Query Language*”. Program ini berjalan sebagai *server* menyediakan *multi-user* mengakses ke sejumlah *database*. *MySQL* umumnya digunakan oleh perangkat lunak bebas yang memerlukan fitur penuh sistem manajemen *database*, seperti *WordPress*, *phpBB* dan perangkat lunak lain yang dibangun pada perangkat lunak *LAMP*. Ia juga digunakan dalam skala sangat tinggi *World Wide Web*, termasuk produk-produk *Google* dan *Facebook*.

c. *PHP*

PHP adalah bahasa pemrograman *script* yang banyak dipakai untuk memrogram situs *web* dinamis, walaupun tidak tertutup

kemungkinan digunakan untuk pemakaian lain. Contoh terkenal dari aplikasi *PHP* adalah *phpBB* dan *MediaWiki* (*software* di belakang *Wikipedia*). Contoh terkenal dari aplikasi *PHP* adalah *phpBB* dan *MediaWiki* (*software* di belakang *Wikipedia*).

5. Bagian-Bagian *XAMPP*

a. *Htdocs*

Htdocs adalah sebuah folder yang digunakan sebagai tempat penyimpanan berkas seperti *PHP*, *HTML*, dan *script* lain yang digunakan dalam sebuah halaman website. Secara kapasitas penyimpanan, *XAMPP* tergantung dari seberapa besar kapasitas *hardisk* di laptop atau komputer anda. Sedangkan bila menggunakan *hosting online*, maka tergantung pilihan waktu membeli sebuah *hosting*.

b. *phpMyadmin*

phpMyadmin adalah sebuah tempat yang digunakan untuk mengelola *database MySQL* yang berada di komputer atau laptop. Untuk mengakses *phpMyadmin* yakni dengan membuka *browser* internet (*Mozilla* atau *chrome*) lalu ketikkan alamat **http://localhost/phpMyadmin** maka akan muncul tampilannya.

c. *Control Panel*

Control Panel adalah sebuah layanan untuk mengelola *XAMPP* baik itu mengontrol (*start* atau *stop XAMPP*) serta layanan *service* lainnya. Secara *online* di dalam *hosting* atau *VPS* dikenal *CPanle*

6. Komponen *XAMPP*

a. *XAMPP 1.8.3 untuk Windows, Termasuk :*

- Apache 2.4.4
- MySQL 6.5.11
- PHP 5.5.0
- phpMyAdmin 4.0.4
- FileZilla FTP Server 0.9.41
- Tomcat 7.0.41 (with mod_proxy_ajp as connector)
- Strawberry Perl 5.16.3.1 Portabel
- XAMPP Control Panel 3.2.1 (dari hackattack142)

b. *XAMPP 1.8.3 untuk Linux, Termasuk :*

- Apache 2.4.4
- MySQL 6.5.11
- PHP 5.5.0
- phpMyAdmin 4.0.4
- OpenSSL 1.0.1e

7. Singkatan *XAMPP*

1. **X** : Program ini dapat dijalankan di banyak sistem operasi, seperti Windows, Linux, Mac OS, dan Solaris.
2. **A** : **Apache**, server aplikasi Web. Apache tugas utama adalah untuk menghasilkan halaman web yang benar kepada pengguna terhadap kode PHP yang sudah dituliskan oleh pembuat halaman web. jika perlu kode PHP juga berdasarkan yang tertulis, dapat database diakses dulu (misalnya MySQL) untuk mendukung halaman web yang dihasilkan.
3. **M** : **MySQL**, server aplikasi database. Pertumbuhannya disebut SQL singkatan dari Structured Query Language. SQL

merupakan bahasa terstruktur yang difungsikan untuk mengolah database. MySQL dapat digunakan untuk membuat dan mengelola database dan isinya. Bisa juga memanfaatkan MySQL guna untuk menambahkan, mengubah, dan menghapus data dalam database.

4. **P : PHP**, bahasa pemrograman web. Bahasa pemrograman PHP adalah bahasa pemrograman untuk membuat web yang server-side scripting. PHP digunakan untuk membuat halaman web dinamis. Sistem manajemen database yang sering digunakan dengan PHP adalah MySQL. namun PHP juga mendukung Pengelolaan sistem database Oracle, Microsoft Access, Interbase, d-base, PostgreSQL, dan sebagainya.
5. **P : Perl**, bahasa pemrograman untuk semua tujuan, pertama kali dikembangkan oleh Larry Wall, mesin Unix. Perl dirilis pertama kali tanggal 18 Desember 1987 yang ditandai dengan keluarnya Perl 1. Pada versi-versi selanjutnya, Perl juga tersedia untuk berbagai sistem operasi Unix (SunOS, Linux, BSD, HP-UX), juga tersedia untuk sistem operasi seperti DOS, Windows, PowerPC, BeOS, VMS, EBCDIC, dan PocketPC.

8. Kelebihan dan Kekurangan *XAMPP*

a. Kelebihan *XAMPP*

- Database Storage Engine ini banyak digunakan oleh programmer apalagi oleh web developer karena sifatnya yang free. Untuk yang expert sudah ada yang bayar.
- Kemampuannya sudah bisa diandalkan, mempunyai kapasitas yang cukup mumpuni sekitar 60.000 tabel dengan jumlah

record mencapai 5.000.000.000 bahkan untuk yang terbaru sudah lebih.

- Keamanan datanya cukup aman walaupun tidak sehebat Postgre apalagi Oracle.
- Engine ini multiplatform sehingga mampu diaplikasikan di berbagai sistem operasi. My Sql cocok diaplikasikan diaplikasi kelas kecil dan menengah.
- Kelebihan paling utama engine ini adalah kecepatannya.

b. Kekurangan *XAMPP*

- Tidak cocok untuk menangani data dengan jumlah yang besar, baik untuk menyimpan data maupun untuk memproses data.
- Memiliki keterbatasan kemampuan kinerja pada server ketika data yang disimpan telah melebihi batas maksimal kemampuan daya tampung server karena tidak menerapkan konsep Technology Cluste.

2.4.3 *MySQL*



Gambar 2.21 XAMPP

1. Pengenalan *MySQL*

MySQL adalah sebuah database management system (manajemen basis data) menggunakan perintah dasar *SQL* (Structured Query

Language) yang cukup terkenal. *Database management system (DBMS)* MySQL multi pengguna dan multi alur ini sudah dipakai oleh banyak orang.

MySQL adalah *DBMS* yang *open source* dengan dua bentuk lisensi, yaitu *Free Software* (perangkat lunak bebas) dan *Shareware* (perangkat lunak berpemilik yang penggunaannya terbatas). Jadi MySQL adalah *database server* yang gratis dengan lisensi *GNU General Public License (GPL)* sehingga dapat Anda pakai untuk keperluan pribadi atau komersil tanpa harus membayar lisensi yang ada.

MySQL masuk ke dalam jenis *RDBMS (Relational Database Management System)*. Maka dari itu, istilah semacam baris, kolom, tabel, dipakai pada MySQL. Contohnya di dalam MySQL sebuah database terdapat satu atau beberapa tabel.

SQL sendiri merupakan suatu bahasa yang dipakai di dalam pengambilan data pada *relational database* atau *database* yang terstruktur. Jadi MySQL adalah *database management system* yang menggunakan bahasa *SQL* sebagai bahasa penghubung antara perangkat lunak aplikasi dengan *database server*.

2. Sejarah MySQL

MySQL adalah pengembangan lanjutan dari proyek UNIREG yang dikerjakan oleh Michael Monty Widenius dan TcX (perusahaan perangkat lunak asal Swedia).

Sayangnya, UNIREG belum terlalu kompatibel dengan database dinamis yang dipakai di website. TcX kemudian mencari alternatif lain dan menemukan perangkat lunak yang dikembangkan oleh David Hughes, yaitu miniSQL atau mSQL. Namun, ditemukan

masalah lagi karena mSQL tidak mendukung indexing sehingga belum sesuai dengan kebutuhan TcX.

Pada akhirnya muncul kerjasama antara pengembang UNIREG (Michael Monty Widenius), mSQL (David Hughes), dan TcX. Kerjasama ini bertujuan untuk mengembangkan sistem database yang baru, dan pada 1995 dirilislah MySQL seperti yang dikenal saat ini. Saat ini pengembangan MySQL berada di bawah Oracle.

3. Kelebihan dan Kekurangan *MySQL*

a. Kelebihan *MySQL*

1. Mendukung Integrasi Dengan Bahasa Pemrograman Lain.

Website atau perangkat lunak terkadang dikembangkan dengan menggunakan berbagai macam bahasa pemrograman, jadi Anda tidak perlu khawatir jika menggunakan MySQL. Maka dari itu, MySQL bisa membantu Anda untuk mengembangkan perangkat lunak yang lebih efektif dan tentu saja lebih mudah dengan integrasi antara bahasa pemrograman.

2. Tidak Membutuhkan RAM Besar

MySQL dapat dipasang pada server dengan spesifikasi kecil. Jadi tidak perlu khawatir jika Anda hanya mempunyai server dengan kapasitas 1 GB karena Anda masih bisa menggunakan MySQL sebagai database Anda.

3. Mendukung Multi User.

MySQL dapat dipakai oleh beberapa user dalam waktu bersamaan tanpa membuatnya crash atau berhenti bekerja. Ini dapat Anda manfaatkan ketika mengerjakan proyek yang sifatnya tim sehingga seluruh tim dapat bekerja dalam waktu bersamaan tanpa harus menunggu user lain selesai.

4. Bersifat Open Source

MySQL adalah sistem manajemen database gratis. Meskipun gratis, bukan berarti database ini mempunyai kinerja buruk. Apalagi lisensi gratis yang dipakai adalah GPL di bawah pengelolaan Oracle sehingga kualitasnya termasuk baik. Selain itu, Anda juga tidak perlu khawatir jika terjadi masalah karena banyak komunitas dan dokumentasi yang membahas soal MySQL.

5. Struktur Tabel yang Fleksibel.

MySQL mempunyai struktur tabel yang mudah dipakai dan fleksibel. Contohnya saat MySQL memproses ALTER TABLE dan lain sebagainya. Jika dibandingkan dengan database lain seperti Oracle dan PostgreSQL, MySQL tergolong lebih mudah.

6. Tipe Data yang Bervariasi.

Kelebihan lain dari MySQL adalah mendukung berbagai macam data yang bisa Anda gunakan di MySQL. Contohnya float, integer, date, char, text, timestamp, double, dan lain sebagainya. Jadi manajemen database sistem ini sangat membantu Anda untuk mengembangkan perangkat lunak yang berguna untuk pengelolaan database di server.

7. Keamanan yang Terjamin

Open source bukan berarti MySQL menyediakan keamanan yang buruk. Malah sebaliknya, MySQL mempunyai fitur keamanan yang cukup apik. Ada beberapa lapisan keamanan yang diterapkan oleh MySQL, seperti level nama host, dan subnetmask. Selain itu MySQL juga dapat mengatur hak akses user dengan enkripsi password tingkat tinggi.

b. Kekurangan *MySQL*

1. Kurang Cocok untuk Aplikasi Game dan Mobile

Anda yang ingin mengembangkan aplikasi game atau perangkat mobile ada baiknya jika mempertimbangkan lagi jika ingin menggunakan MySQL. Kebanyakan pengembang game maupun aplikasi mobile tidak menggunakan karena memang database manajemen sistem ini masih kurang bagus dipakai untuk sistem aplikasi tersebut.

2. Sulit Mengelola Database yang Besar

Jika Anda ingin mengembangkan aplikasi atau sistem di perusahaan dengan database yang cukup besar, ada baiknya jika menggunakan database manajemen sistem selain MySQL. MySQL dikembangkan supaya ramah dengan perangkat yang mempunyai spesifikasi rendah, itulah mengapa MySQL tidak memiliki fitur yang lengkap seperti aplikasi lainnya.

3. Technical Support yang Kurang Bagus

Sifatnya yang open source terkadang membuat aplikasi tidak menyediakan technical support yang memadai. Technical support MySQL diklaim kurang bagus. Hal ini membuat pengguna kesulitan. Apalagi jika pengguna mengalami masalah

yang berhubungan dengan pengoperasian perangkat lunak tersebut dan membutuhkan bantuan technical support.

BAB III

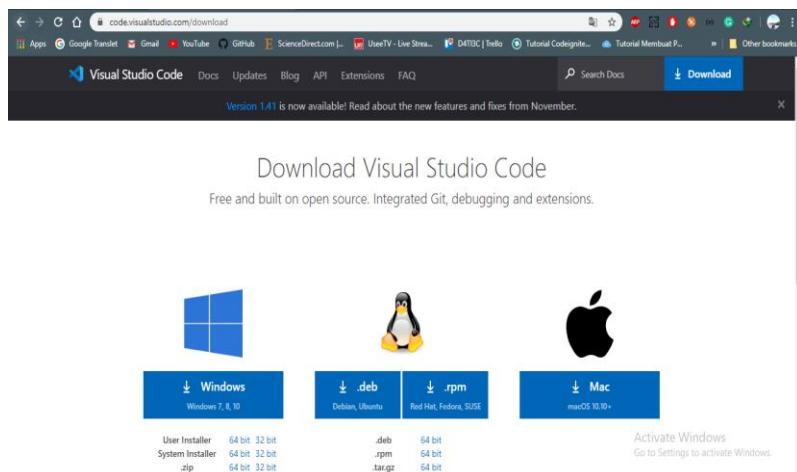
INSTALASI *TOOLS* YANG DIGUNAKAN

Ada beberapa *tools* yang digunakan dalam pembuatan sistem informasi ini. Pada bab kali ini penulis akan menjelaskan secara rinci bagaimana proses instalasi pada setiap *tools* yang akan digunakan.

3.1 Visual Studio Code

Pada bab sebelumnya telah dijelaskan apa itu *visual studio code*. *Visual studio code* merupakan sebuah teks editor yang digunakan dalam pembuatan sistem infromasi ini. Berikut penulis akan menjelaskan langkah-langkah instalasi *visual studio code*.

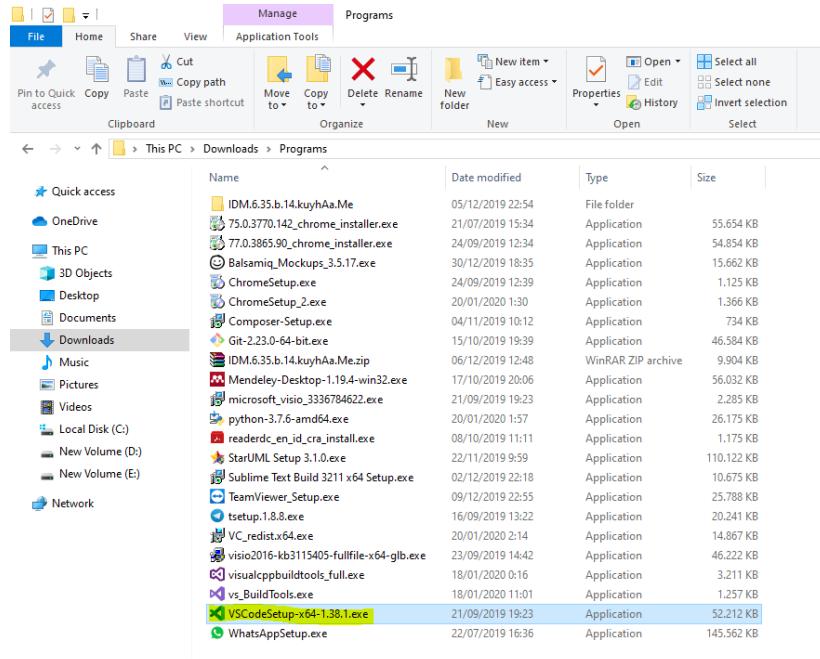
1. Pada tahapan awal yaitu kita terlebih dahulu harus *download software visual studio code*, di website resmi *visual studio code* yaitu: <https://code.visualstudio.com/download>



Gambar 3.1 Download Visual Studio Code

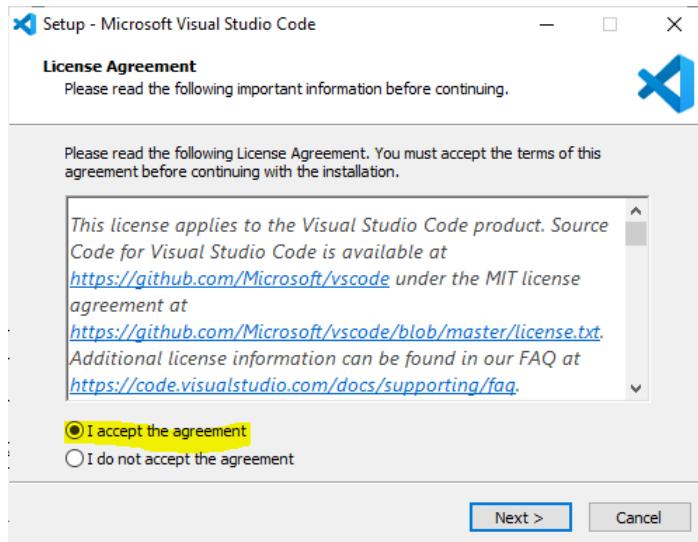
Pada gambar di atas bisa dilihat bahwa pada website *visual studio code* mereka menyediakan *software* tersebut untuk berbagai jenis sistem operasi seperti: *windows*, *linux* dan *machintos*.

2. Pada tahapan kedua adalah buka *directory* dimana *file download* tesimpan.



Gambar 3.2 Directory File Download

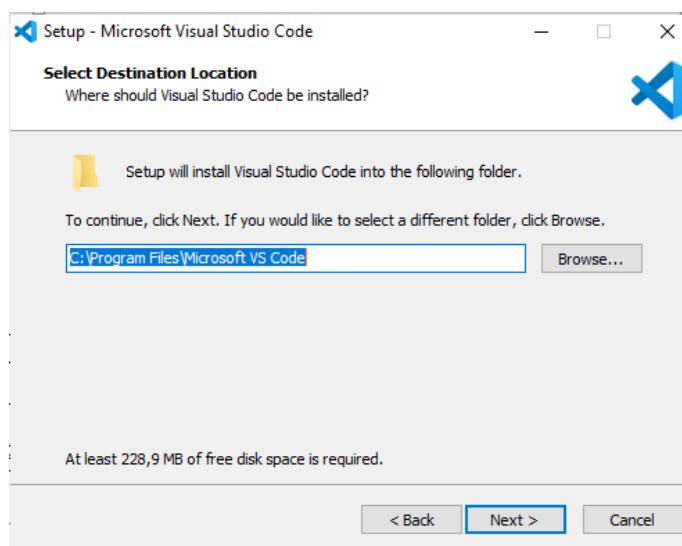
3. Pada tahap selanjutnya *double klik* pada *file visual studio code* yang telah didownload.



Gambar 3.3 Proses instalasi visual studio code (1)

Pada gambar di atas adalah tahap pertama proses instalasi *visual studio code*. Pada tahap ini kita diminta untuk menyetujui persetujuan yang disediakan oleh *visual studio code*. Pada proses ini kita hanya perlu untuk klik “*I accept the agreement*”, setelah itu klik *next*.

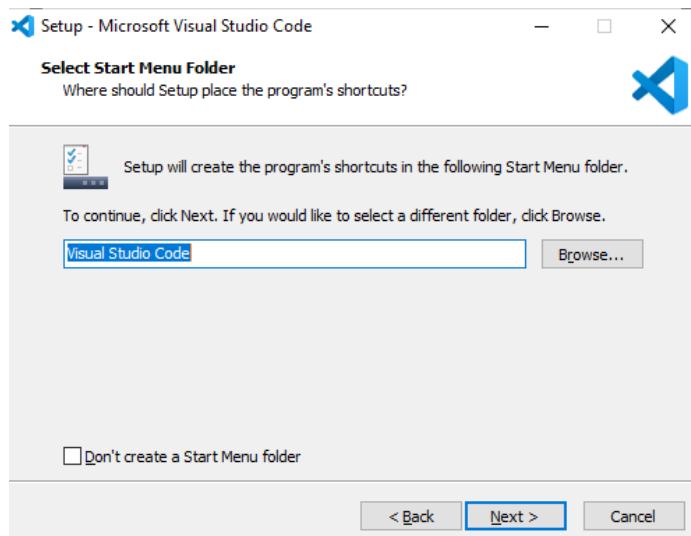
4. Pada tahap selanjutnya adalah atur tempat penyimpanan file instalasi pada perangkat anda.



Gambar 3.4 Proses instalasi visual studio code (2)

Pada gambar di atas saya menyimpan file instalasi *visual studio code* pada folder C. Jika sudah menentukan tempat penyimpanan file instalasi klik *button next*.

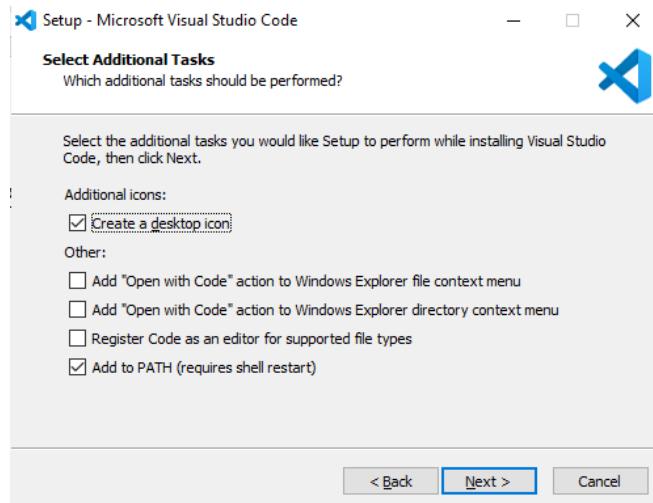
5. Pada tahapan selanjutnya program akan membuat *shorcut* aplikasi.



Gambar 3.5 Proses instalasi visual studio code (3)

Jika sudah klik *next*.

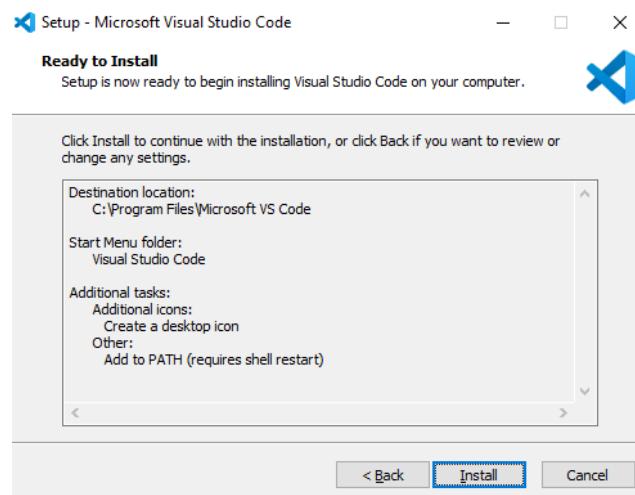
6. Pada tahapan selanjutnya kita diminta untuk memilih apakah kita ingin membuat *shortcut* aplikasi di *desktop* serta banyak pilihan yang lainnya.



Gambar 3.6 Proses instalasi visual studio code (4)

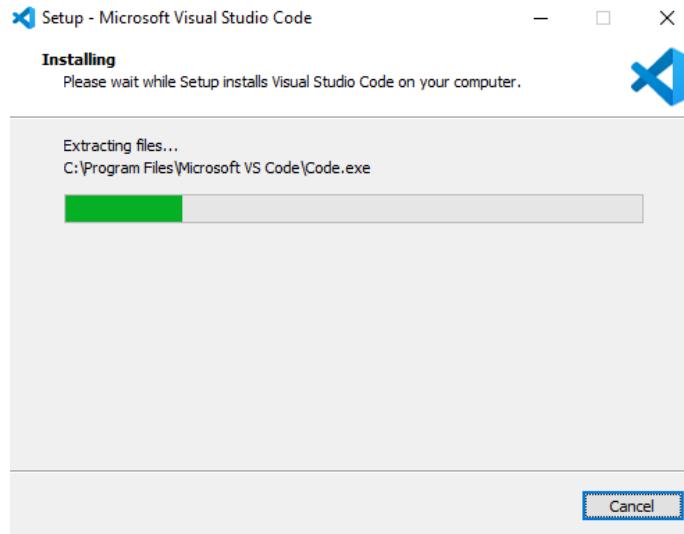
Pada gambar di atas saya memilih untuk membuat *shorcut* aplikasi pada *desktop*. Kemudian klik *next*.

7. Pada tahapan selanjutnya adalah proses instalasi. Kemudian klik *install*.



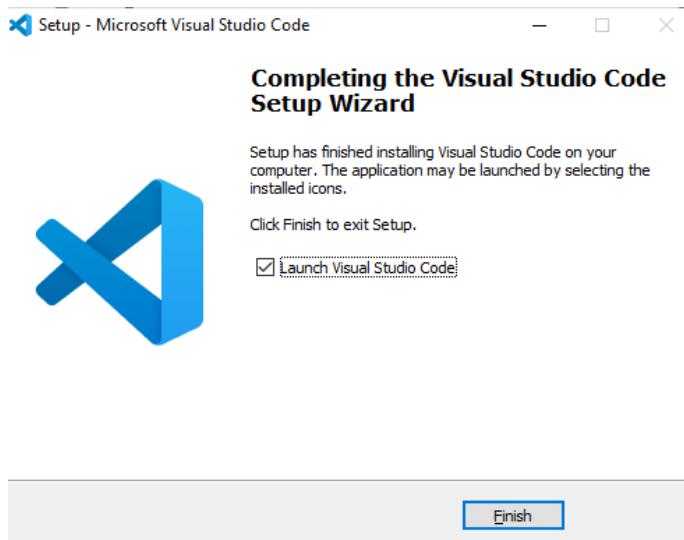
Gambar 3.7 Proses instalasi visual studio code (5)

8. Jika sudah klik *install* maka akan tampil proses seperti gambar di bawah ini, dimana *visual studio code* dalam proses instalasi.



Gambar 3.8 Proses instalasi visual studio code (6)

9. Jika proses di atas sudah selesai, maka akan tampil gambar seperti di bawah ini:



Gambar 3.9 Proses instalasi visual studio code (7)

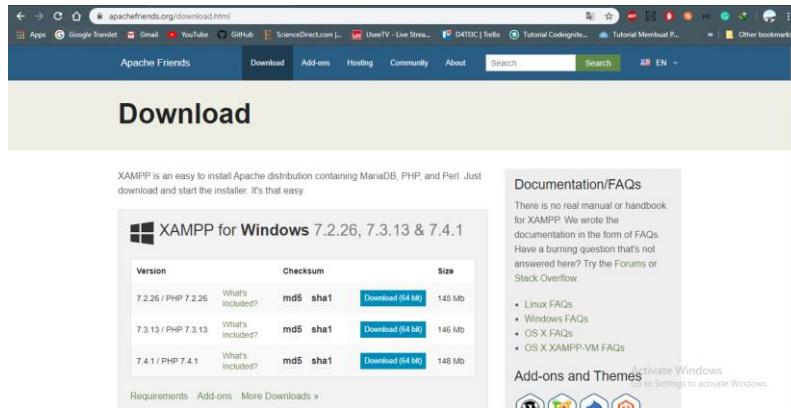
Kemudian klik *finish*, maka *visual studio code* sudah bisa digunakan.

3.2 XAMPP

Tools yang kedua adalah *XAMPP*. Berikut penulis akan menjelaskan langkah-langkah instalasi dari *XAMPP*:

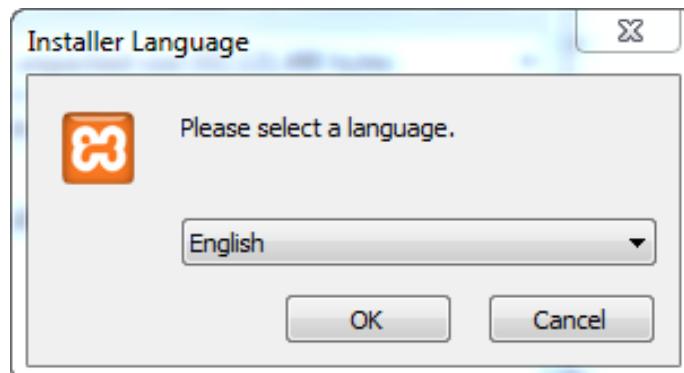
3.2.1 Instalasi *XAMPP*

1. Untuk langkah pertama yang dilakukan dalam instalasi *XAMPP* adalah kita harus *download* *XAMPP* terlebih dahulu. Kita bisa download pada link ini: <https://www.apachefriends.org/download.html>



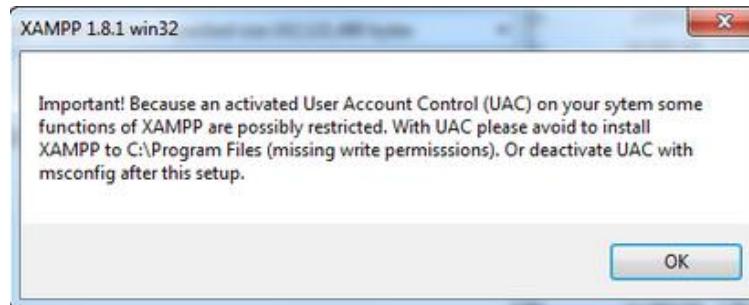
Gambar 3.10 Download *XAMPP*

2. *Double klik* file *XAMPP* yang telah di *download*. Selanjutnya akan muncul jendela “*installer language*” seperti di bawah ini:

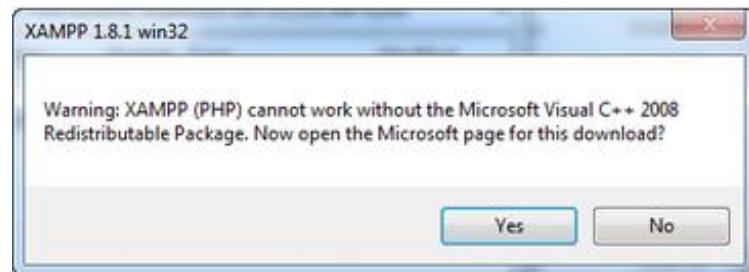


Gambar 3.11 Proses Instalasi *XAMPP* (1)

3. Kadang pada proses ini akan muncul pesan eror. Jika ada, abaikan saja. Kemudian lanjutkan dengan klik OK dan YES.



Gambar 3.12 Proses Instalasi XAMPP (2)



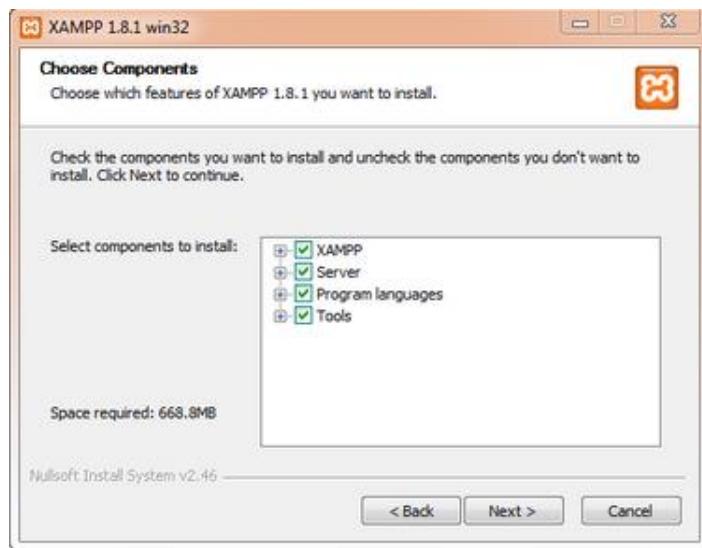
Gambar 3.13 Proses Instalasi XAMPP (3)

4. Berikutnya akan muncul jendela yang isinya meminta anda untuk menutup semua aplikasi yang sedang berjalan. Jika semua aplikasi sudah ditutup, klik tombol *next*.



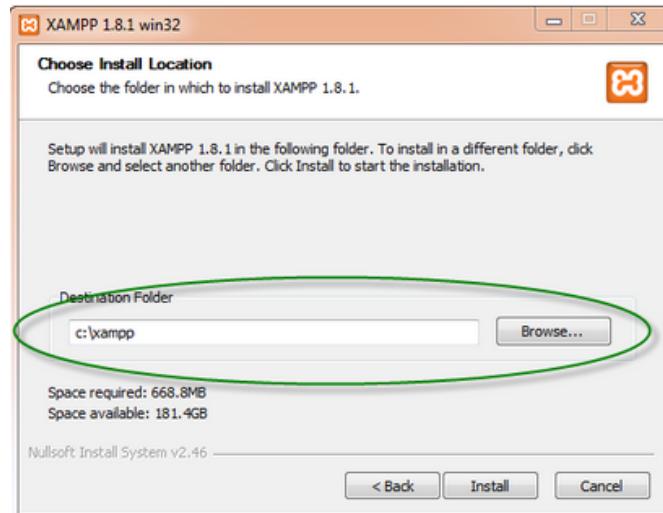
Gambar 3.14 Proses Instalasi XAMPP (4)

5. Selanjutnya anda akan diminta untuk memilih aplikasi yang mau diinstal. Centang saja semua pilihan dan klik tombol *next*.



Gambar 3.15 Proses Instalasi XAMPP (5)

6. Kemudian anda akan diminta untuk menentukan lokasi folder penyimpanan file-file dan folder XAMPP. Scara *default* akan diarahkan ke lokasi c:\xampp. Namun jika anda ingin menyimpannya pada folder lain bisa klik *browse* dan tentukan tempat penyimpanan yang anda inginkan. Jika sudah selesai, lanjutkan dan klik tombol *install*.



Gambar 3.16 Proses Instalasi XAMPP (6)

7. Tunggu hingga proses instalasi selesai. Jika sudah muncul jendela seperti gambar di bawah ini, klik tombol *finish* untuk menyelesaiakannya.



Gambar 3.17 Proses Instalasi XAMPP (7)

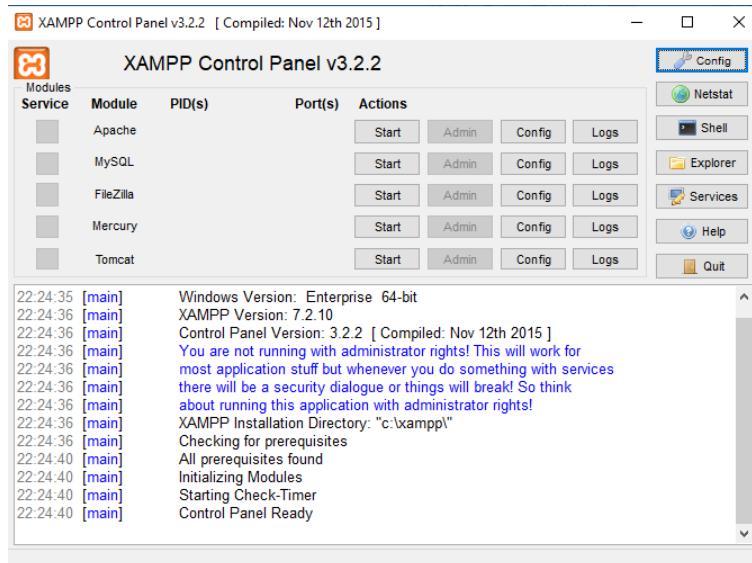
8. Berikutnya akan muncul jendela dialog seperti gambar di bawah ini yang mananyakan anda apakah langsung menjalankan aplikasi XAMPP atau tidak. Jika ya, maka klik YES.



Gambar 3.18 Proses Instalasi XAMPP (8)

3.2.2 Cara Menjalankan Aplikasi XAMPP

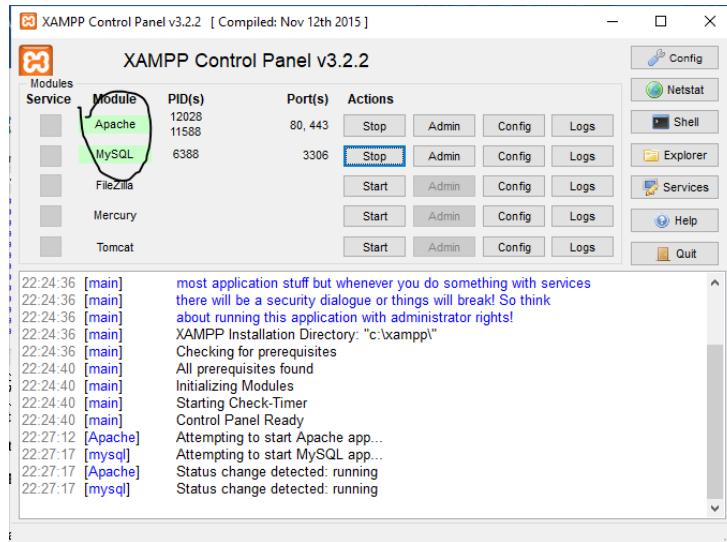
1. Bukalah aplikasi XAMPP, bisa melalui *Start Menu* atau *desktop*, dan klik ikon XAMPP. Atau jika anda membukannya begitu proses instalasi maka klik *YES*, seperti gambar pada proses instalasi sebelumnya.



Gambar 3.19 Menjalankan Aplikasi XAMPP (1)

2. Setelah terbuka silahkan klik tombol *start* pada kolom *action* sehingga tombol tersebut berubah menjadi *stop* atau berwarna hijau. Artinya itulah aplikasi yang dijalankan. Biasanya jika saya menggunakan XAMPP yang saya jalankan hanyalah aplikasi

Apache dan *MySQL*, karena saya tidak membutuhkan aplikasi lain seperti *Filezilla*, dan lain-lain.



Gambar 3.20 Menjalankan Aplikasi XAMPP (2)

3. Sekarang buka *browser* yang anda gunakan pada perangkat anda. Kemudia coba ketikkan <http://localhost/xampp> pada *address bar*. Jika sudah akan muncul gambar seperti di bawah ini, berarti instalasi XAMPP pada perangkat anda telah berhasil.

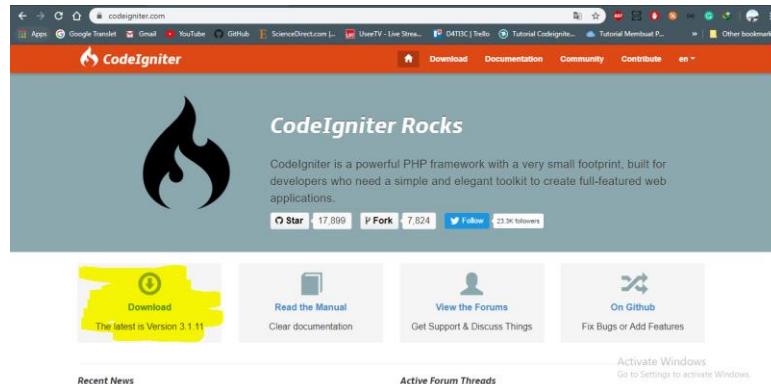


Gambar 3.21 Menjalankan Aplikasi XAMPP (3)

3.3 CodeIgniter (CI)

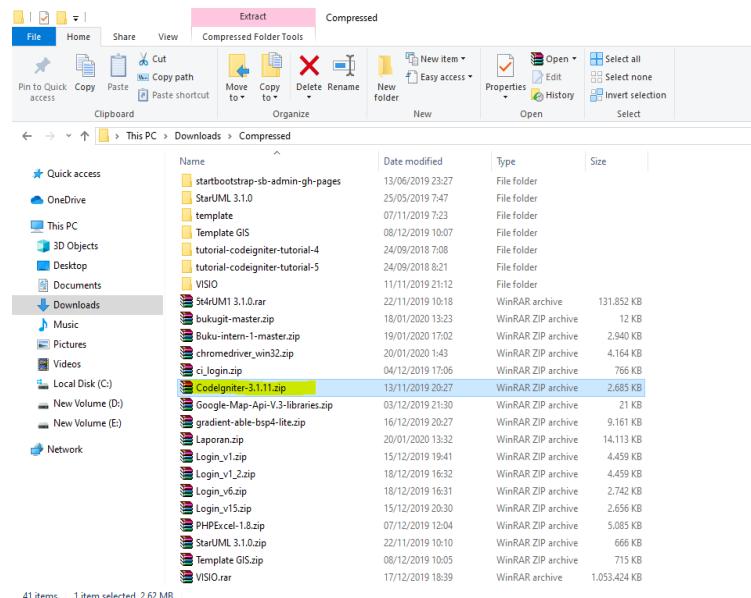
Tools yang ketiga adalah *framework codeigniter*. Berikut penulis akan menjelaskan langkah-langkah instalasi dari *codeigniter*:

1. Pada tahap pertama *download file codeigniter* terlebih dahulu. Anda bisa mendownloadnya pada website resmi *codeigniter* <https://codeigniter.com/>



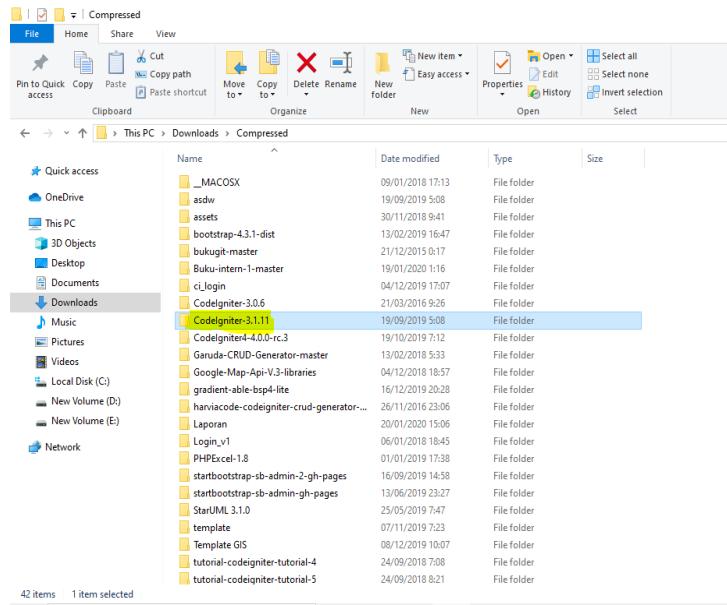
Gambar 3.22 Download CodeIgniter

2. Setelah melakukan proses *download* buka tempat penyimpanan hasil download pada perangkat anda.

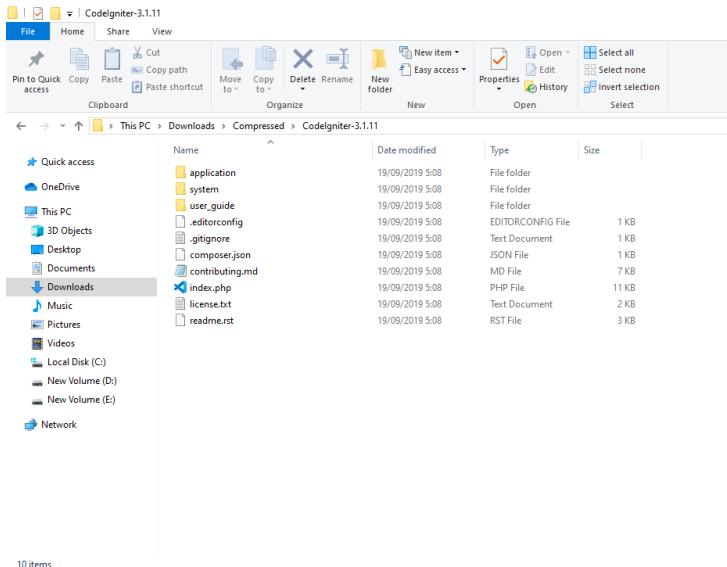


Gambar 3.23 File CodeIgniter yang telah di download

3. Ekstrak file codeigniter yang telah di download.

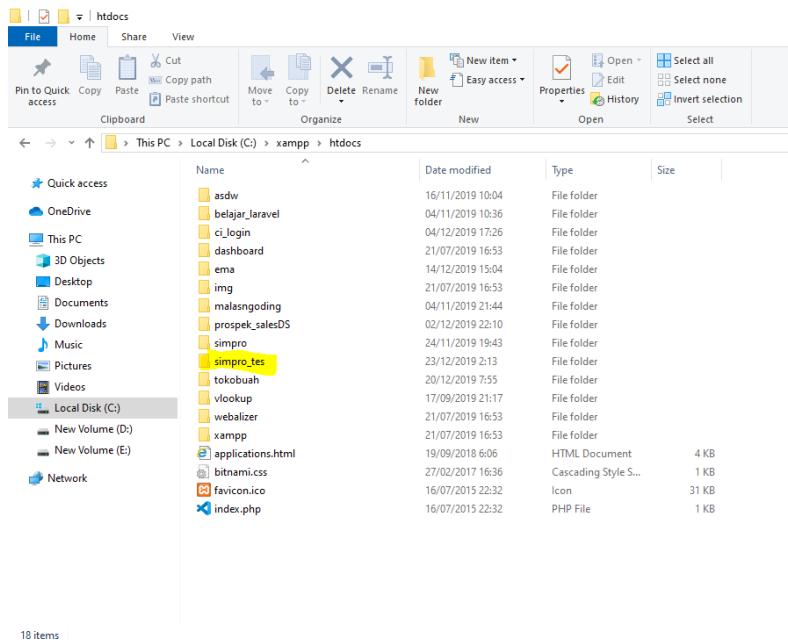


Gambar 3.24 Hasil Ekstrak File CodeIgniter (1)



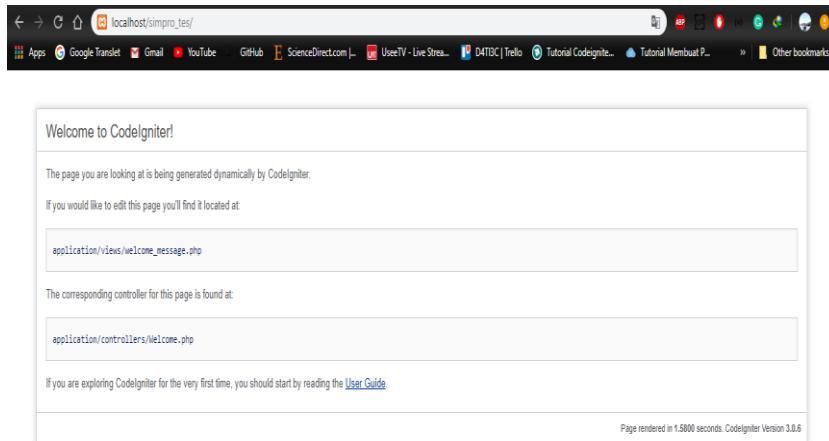
Gambar 3.25 Hasil Ekstrak File CodeIgniter (2)

- Setelah di ekstrak, kemudian pindahkan folder *codeigniter* tersebut ke folder C:\xampp\htdocs. Kemudian ganti nama folder tersebut sesuai kebutuhan dan keinginan anda masing-masing.



Gambar 3.26 Folder CodeIgniter yang telah di rename

5. Kemudian buka *browser* yang ada di perangkat anda kemudian ketikkan http://localhost/simpro_tes jika instalasi *CodeIgniter* berhasil maka akan muncul gambar di bawah ini:



Gambar 3.27 Hasil Instalasi CodeIgniter

BAB IV

ANALISIS DAN PERANCANGAN PEMBUATAN SISTEM INFORMASI *APPROVAL*

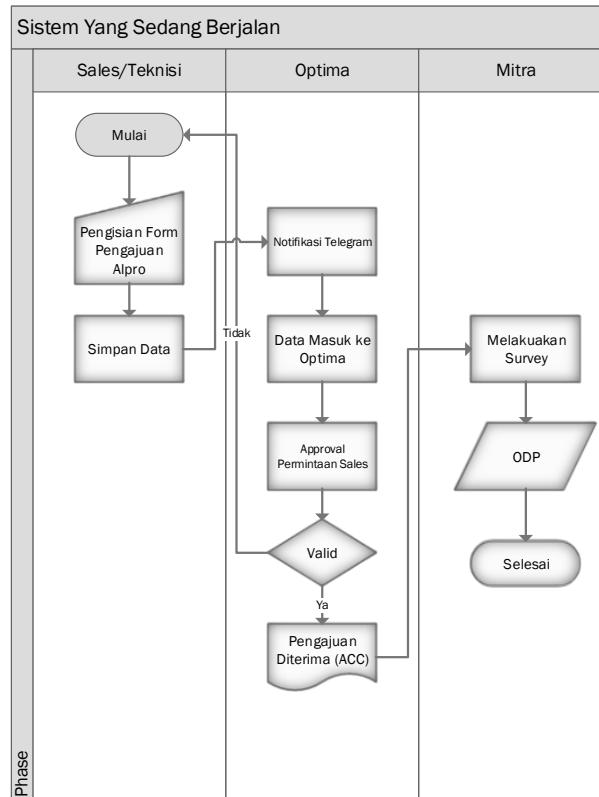
Pada bab ini penulis akan menjelaskan tentang perancangan dari pembuatan sistem infromasi *approval* ini. Apa saja yang dibutuhkan dalam membangun sistem informasi *approval*.

4.1 Perancangan Sistem Informasi

Dalam membangun suatu aplikasi atau sistem informasi sangat diperlukan suatu perancangan, supaya aplikasi atau sistem infromasi yang di buat sesuai dengan yang diinginkan, maka dari itu diperlukan suatu perancangan sistem seperti berikut :

4.1.1 Analisis Sistem Yang Sedang Berjalan

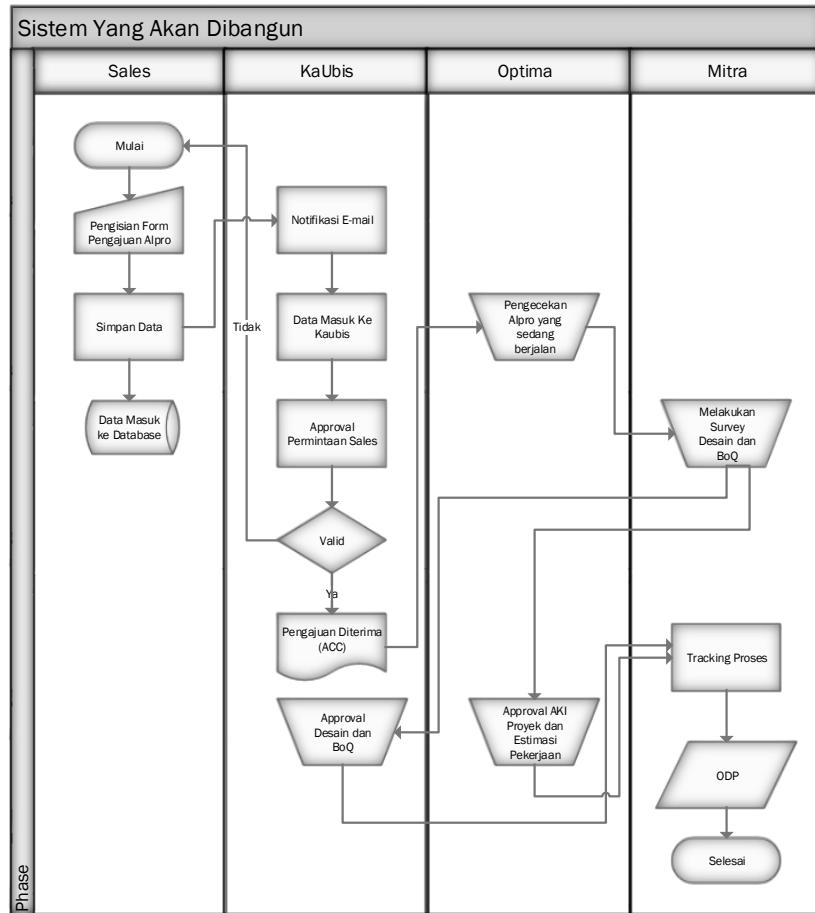
Tahapan yang diperlukan dalam pembuatan suatu program yaitu menganalisa sistem yang telah ada, dimana analisa sistem akan memberikan gambaran tentang sistem yang saat ini sedang berjalan dan bertujuan mengetahui lebih jelas bagaimana cara kerja atau rancangan sistem tersebut serta untuk mendefinisikan dan mengevaluasi permasalahan terjadi dan kebutuhan-kebutuhan yang diharapkan dapat diusulkan suatu perbaikan.



Gambar 4.1 Flowmap Sistem Yang Sedang Berjalan

4.1.2 Analisis Sistem Yang Akan Dibangun

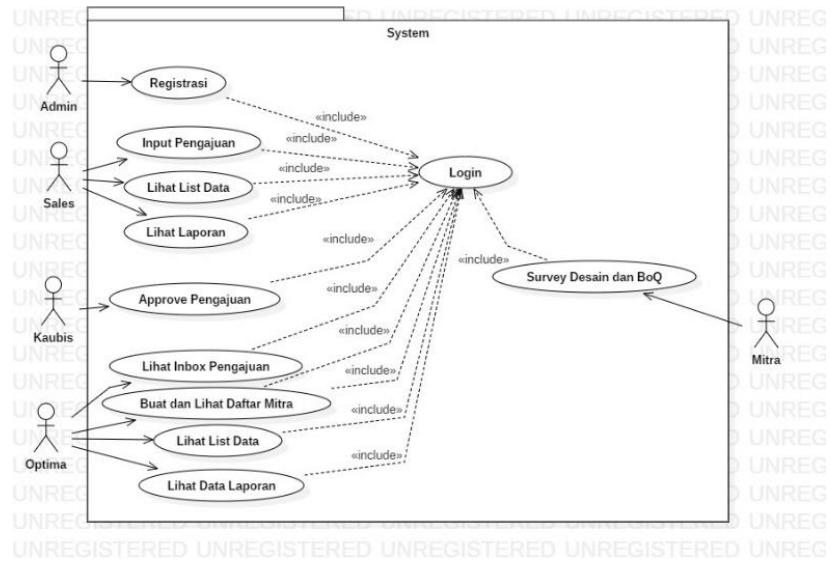
Setelah melakukan analisa dan mengetahui kelemahan kelemahan pada sistem yang sedang berjalan, maka dapat dibuat sebuah sistem baru yang dapat memperbaiki kekurangan kekurangan terhadap sistem yang lama sehingga dapat membantu untuk memproses informasi dengan lebih cepat.



Gambar 4.2 Flowmap Sistem Yang Akan Dibangun

4.1.3 Use Case Diagram

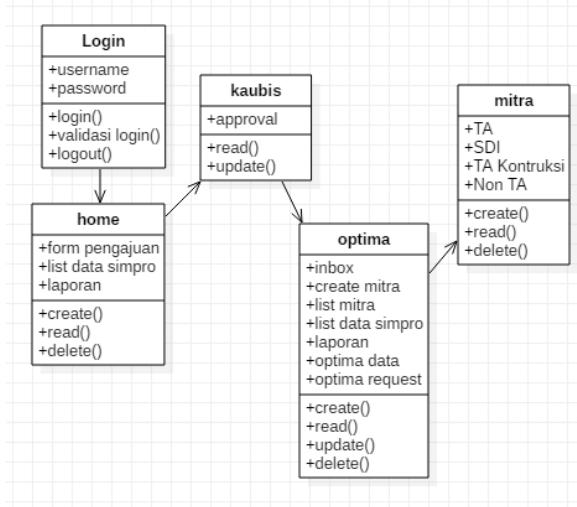
Diagram *use case* menggambarkan interaksi antar *use case* dan *actor* dalam suatu sistem.



Gambar 4.3 UseCase Diagram

4.1.4 Class Diagram

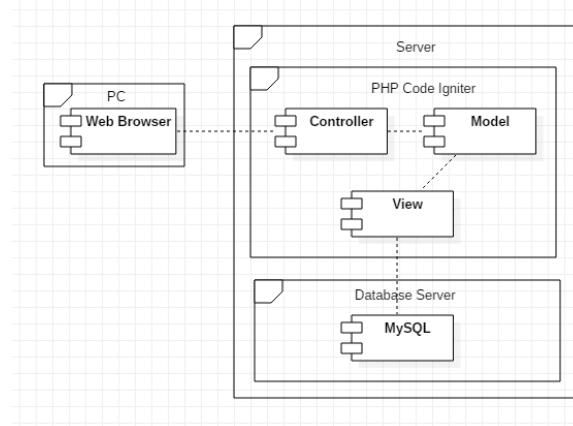
Class diagram berfungsi untuk menggambarkan suatu struktur dan hubungan objek-objek yang ada pada sistem. Struktur itu meliputi atribut-atribut dan method-method yang ada pada masing-masing kelas. Adapun Class Diagram pada sistem informasi *approval* ini yaitu sebagai berikut :



Gambar 4.4 Class Diagram

4.1.5 Component Diagram

Component diagram adalah unit fisik nyata yang menjadi bagian dari *deployment independent*. *Component* ini diimplementasikan meskipun pada sistem yang kecil.



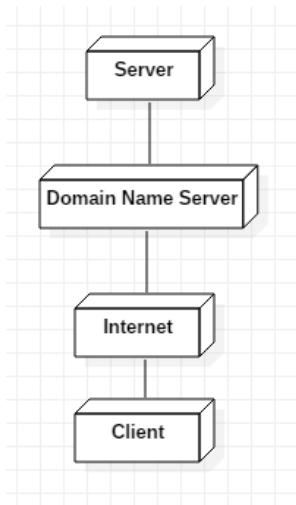
Gambar 4.5 Component Diagram

4.1.6 Deployment Diagram

Deployment diagram merupakan suatu tampilan atau kinerja diri sebuah sistem yang baru sesuai dengan perancangan data yang diambil dari beberapa objek.

4.1.6.1 Deployment Diagram Hardware

Deployment diagram ini menunjukkan *hardware* yang digunakan pada jejaring kantor yang kecil. *Server* terhubung dengan *domain name server* dan *internet* serta *client (windows)*.



Gambar 4.6 Deployment Diagram Hardware

4.1.6.2 Deployment Diagram Software

Deployment diagram ini menunjukkan *software* yang digunakan pada jejaring kantor yang kecil. Database MySQL terhubung dengan *Web Browser*.



Gambar 4.7 Deployment Diagram Software

4.2 Perancangan Database

Basis data (*database*) merupakan salah satu komponen yang penting dalam pembuatan sistem informasi, karena basis data merupakan hal pokok dalam menyediakan informasi tentang data kepada para pengguna khususnya.

Rancangan *database* terdiri dari beberapa tabel yang saling berhubungan. Dalam pembuatan *database* ini, digunakan MySQL sebagai *database server*. Berikut adalah desain *database server*:

Tabel 4.1 Perancangan Database *tbl_sales*:

Field	Type	Null	Key
<i>id_simpro</i>	<i>varchar(15)</i>	<i>NO</i>	<i>Primary</i>
<i>tanggal</i>	<i>date</i>	<i>NO</i>	
<i>witel</i>	<i>varchar(25)</i>	<i>NO</i>	
<i>datel</i>	<i>varchar(10)</i>	<i>NO</i>	
<i>sto</i>	<i>varchar(10)</i>	<i>NO</i>	
<i>latitude</i>	<i>varchar(30)</i>	<i>NO</i>	
<i>longitude</i>	<i>varchar(10)</i>	<i>NO</i>	
<i>unsc</i>	<i>varchar(10)</i>	<i>NO</i>	
<i>qty_odp</i>	<i>int(2)</i>	<i>NO</i>	
<i>alamat</i>	<i>varchar(50)</i>	<i>NO</i>	
<i>keterangan</i>	<i>varchar(10)</i>	<i>NO</i>	
<i>mitra</i>	<i>varchar(30)</i>	<i>NO</i>	
<i>project_name</i>	<i>varchar(30)</i>	<i>NO</i>	
<i>survey_status</i>	<i>varchar(30)</i>	<i>NO</i>	
<i>status_fisik</i>	<i>varchar(30)</i>	<i>NO</i>	
<i>odp_name</i>	<i>varchar(30)</i>	<i>NO</i>	
<i>status_1</i>	<i>varchar(30)</i>	<i>NO</i>	
<i>status</i>	<i>varchar(30)</i>	<i>NO</i>	
<i>created</i>	<i>varchar(20)</i>	<i>NO</i>	

Tabel 4.2 Perancangan Database *tbl_mitra*:

Field	Type	Null	Key	Extra
<i>id_mitra</i>	<i>int(10)</i>	<i>NO</i>	<i>Primary</i>	<i>Auto_Increment</i>
<i>nama_mit</i>	<i>varchar(30)</i>	<i>NO</i>		

Tabel 4.3 Perancangan Database *tbl_login*:

Field	Type	Null	Key	Extra
<i>id_user</i>	<i>int(10)</i>	<i>NO</i>	<i>Primary</i>	<i>Auto_Increment</i>
<i>username</i>	<i>varchar(30)</i>	<i>NO</i>		
<i>password</i>	<i>varchar(255)</i>			
<i>Level</i>	<i>enum</i>	<i>Yes</i>		
		<i>('1','2','3','4')</i>		
<i>status</i>	<i>int (11)</i>	<i>NO</i>		
<i>tgl</i>	<i>datetime</i>	<i>NO</i>		

Tabel 4.4 Perancangan Database *tbl_optimarequest*:

Field	Type	Null	Key	Extra
<i>id_request</i>	<i>int(11)</i>	<i>NO</i>	<i>Primary</i>	<i>Auto_Increment</i>
<i>sto</i>	<i>varchar(100)</i>	<i>NO</i>		
<i>lokasi</i>	<i>varchar(100)</i>	<i>NO</i>		

Tabel 4.5 Perancangan Database *tbl_sdi*:

Field	Type	Null	Key	Extra
<i>id_boq</i>	<i>int(11)</i>	<i>NO</i>	<i>Primary</i>	<i>Auto_Increment</i>
<i>pekerjaan</i>	<i>varchar(100)</i>	<i>NO</i>		
<i>sto</i>	<i>varchar(100)</i>	<i>NO</i>		
<i>lokasi</i>	<i>varchar(100)</i>	<i>NO</i>		
<i>desigimator</i>	<i>enum</i>	<i>NO</i>		
<i>uraian_pekerjaan</i>	<i>enum</i>	<i>NO</i>		
<i>satuan</i>	<i>enum</i>	<i>NO</i>		
<i>status</i>	<i>enum</i>	<i>NO</i>		

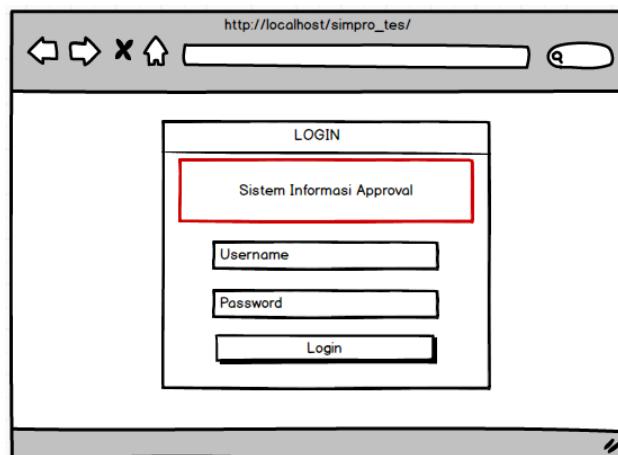
Tabel 4.6 Perancangan *Database* *tbl_ta*:

<i>Field</i>	<i>Type</i>	<i>Null</i>	<i>Key</i>	<i>Extra</i>
<i>id_survey</i>	<i>int(11)</i>	<i>NO</i>	<i>Primary</i>	<i>Auto_Increment</i>
<i>sto</i>	<i>varchar(100)</i>	<i>NO</i>		
<i>lokasi</i>	<i>varchar(100)</i>	<i>NO</i>		
<i>gambar_olt</i>	<i>varchar(100)</i>	<i>NO</i>		
<i>gambar_ftm</i>	<i>varchar(100)</i>	<i>NO</i>		
<i>gambar_odc</i>	<i>varchar(100)</i>	<i>NO</i>		
<i>gambar_odp</i>	<i>varchar(100)</i>	<i>NO</i>		
<i>status</i>	<i>enum</i>	<i>NO</i>		

4.3 Perancangan *User Interface*

Rancangan *Interface* adalah rancangan pembangunan dari komunikasi antar pemakai (administrator) dengan komputer. Antar muka (*interface*) ini terdiri dari proses pemasukan data ke sistem dan menampilkan *output* informasi kepada administrator. Berikut beberapa gambar bentuk rancangan *user interface* dari sistem informasi *approval*:

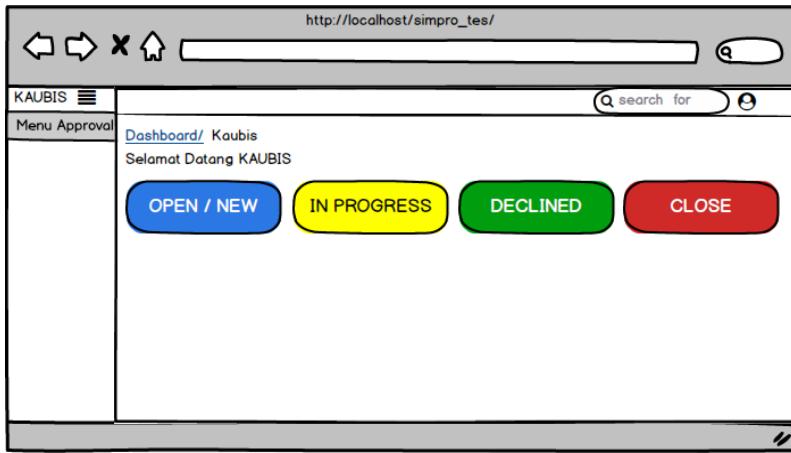
1. Menu *Login*



Gambar 4.8 Menu *Login*

Pada Menu *Login* di atas terdapat beberapa fitur, yaitu label, *text*, dan *button*. Dimana *user* akan mengisi *Username* dan *Password* untuk bisa masuk ke menu atau *form* selanjutnya. Kemudian *user* akan menekan *button login* maka secara otomatis akan dieksekusi oleh *database* dan masuk ke menu selanjutnya.

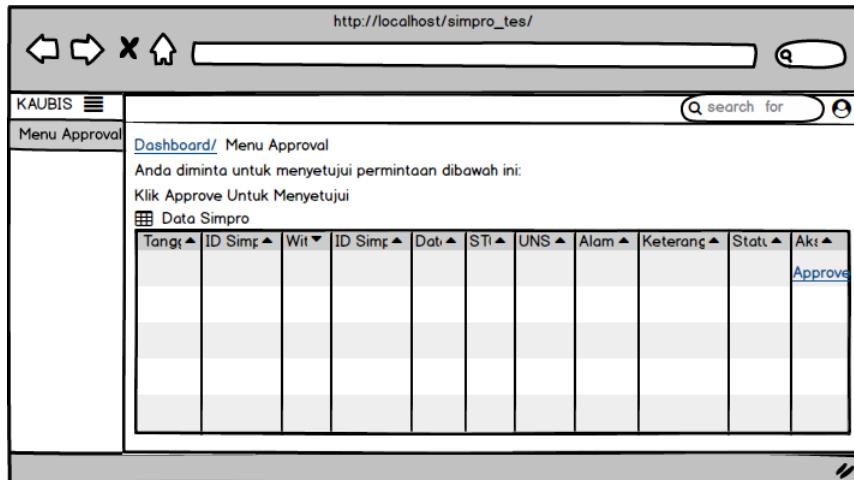
2. *Dashboard User I*



Gambar 4.9 Dashboard User I

Gambar sebelumnya adalah *dashboard* dari *User I*. Tampilan ini akan tampil setelah *user login*. Secara otomatis sistem akan mengarahkan *user* pada menu ini.

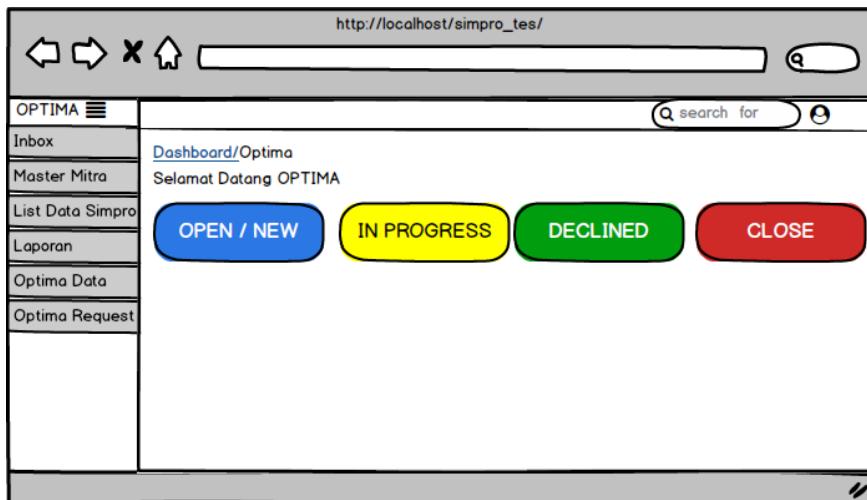
3. Menu *Approve User I*



Gambar 4.10 Menu Approval User I

Gambar di atas merupakan menu approval dari *user I*. Dimana data yang diinputkan oleh *user* sebelumnya akan otomatis masuk ke menu ini. Pada menu ini *use I* diminta untuk menyetujui permintaan yang masuk.

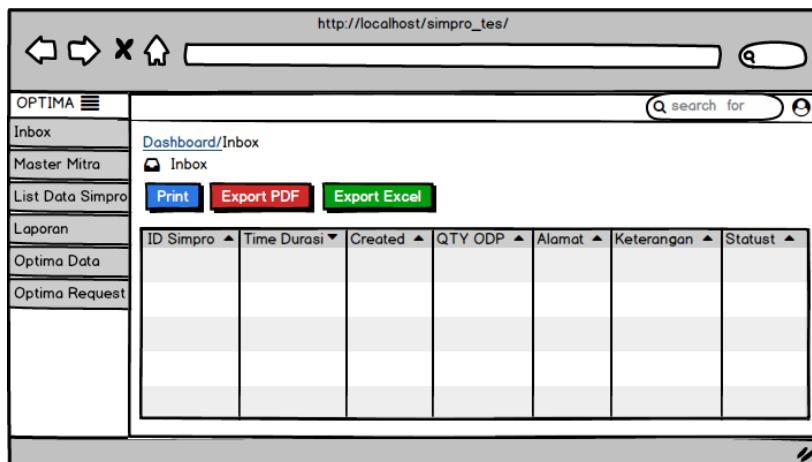
4. Dashboard User II



Gambar 4.11 Dashboard User II

Gambar sebelumnya adalah *dashboard* dari *User II*. Tampilan ini akan tampil setelah *user login*. Secara otomatis sistem akan mengarahkan user pada menu ini.

5. Menu *Inbox User II*



*Gambar 4.11 Menu *Inbox User II**

Gambar di atas merupakan menu *inbox* pada *user II*. Pada menu ini data yang telah diinputkan oleh *user* sebelumnya akan masuk ke menu ini.

6. Menu *Master Mitra (User II)*

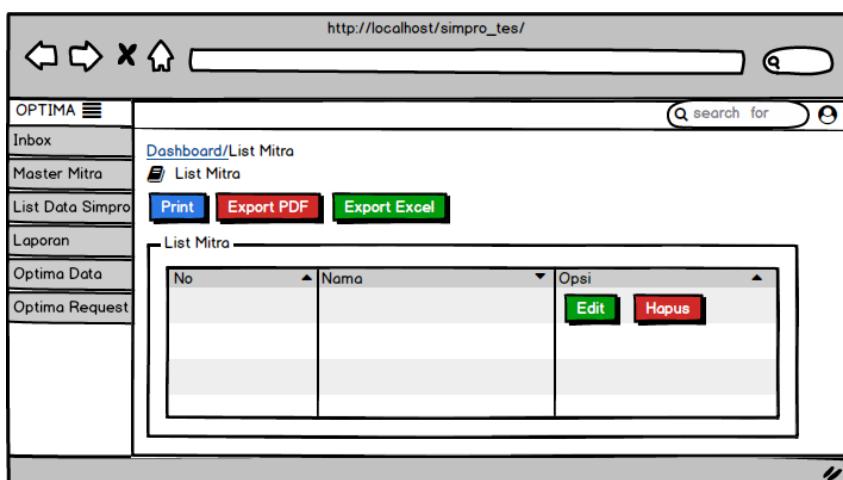
a. *Create Mitra*



Gambar 4.12 Menu Create Mitra User II

Gambar di atas merupakan submenu *create* mitra pada menu master di *user II*. Pada menu ini *user II* bisa membuat mitra baru. Di dalam menu ini terdapat beberapa fitur yaitu *text input* dan 2 *button*. Dimana *text input* digunakan untuk input data mitra yang baru jika sudah data mitra yang telah dibuat bisa disimpan dengan klik *button* simpan data. Jika mitra yang diinputkan salah sebelum menekan tombol simpan data, data tersebut bisa di *reset*. Kemudian bisa menginputkan data yang baru lagi.

b. *List Mitra*

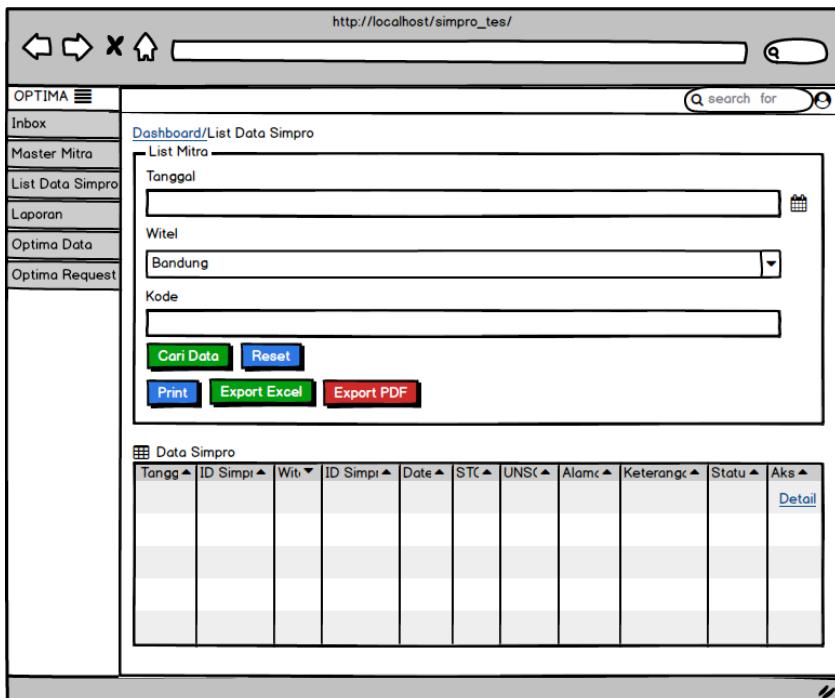


Gambar 4.13 Menu List Mitra User II

Gambar di atas merupakan submenu *list* mitra pada menu master pada *user II*. Daftar mitra yang dibuat pada menu *create* mitra akan masuk ke menu *list* mitra. Pada menu *list* mitra ada beberapa fitur yaitu: *table* untuk menampilkan data dari *create* mitra, kemudian ada 2 *button* yaitu *button* edit dan hapus. Pada menu ini kita bisa melakukan 2 fungsi yaitu edit dan hapus. Dimana data yang masuk dari menu *create* mitra bisa diedit dan dihapus. Jika kita menekan *button* edit, maka sistem akan mengarahkan *user* ke menu *create* mitra dan *user* bisa mengubah nama

mitra sebelumnya menjadi nama mitra yang baru. Jika mitra sudah diperlukan lagi *user* bisa menghapus data dengan menekan *button* hapus, secara otomatis data akan dihapus dari menu *list* mitra dan *database*.

7. Menu List Data

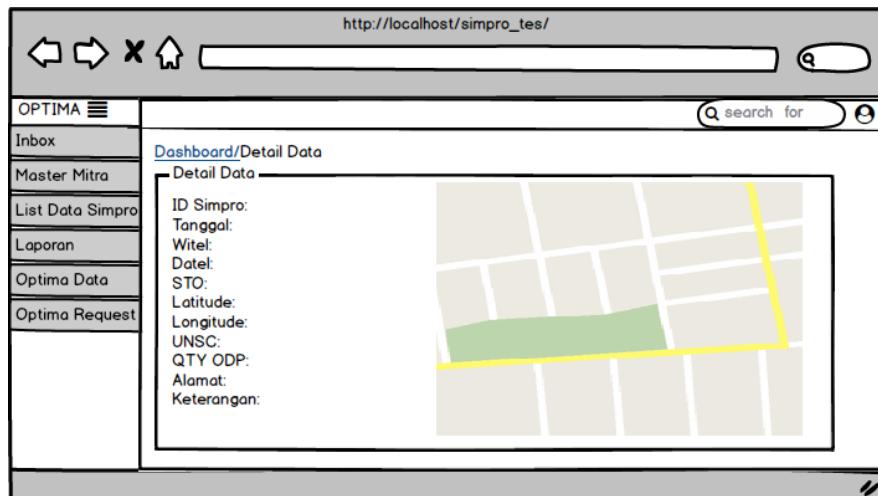


The screenshot shows a web application interface for 'List Data Simpro'. On the left, a vertical menu bar is visible with the following items: OPTIMA (with a dropdown arrow), Inbox, Master Mitra, List Data Simpro, Laporan, Optima Data, and Optima Request. The main content area has a header 'Dashboard/List Data Simpro' with a search bar. Below the header, there are search fields for 'Tanggal' (Date) and 'Witel' (Area), with a dropdown menu showing 'Bandung'. There is also a 'Kode' (Code) field. At the bottom of the search area are three buttons: 'Cari Data' (Search Data) in green, 'Reset' in blue, and 'Print', 'Export Excel', and 'Export PDF' in red. Below the search area is a table titled 'Data Simpro' with columns: Tanggal, ID Simpro, Witel, ID Simpro, Date, STC, UNSC, Alamat, Keterangan, Status, and Aksi. The 'Aksi' column contains a 'Detail' link for each row. The URL in the browser's address bar is 'http://localhost/simpro_tes/'.

Gambar 4.15 Menu List Data User II

Gambar di atas merupakan menu yang menampilkan data yang telah diinputkan oleh *user* sebelumnya. Yang ditampilkan dalam bentuk tabel, dimana salah satu *field* dari tabel yang ditampilkan ada detail data yang jika di klik akan menampilkan detail data yang ada di setiap *field* tabel. Pada menu ini kita juga bisa melakukan pencarian data berdasarkan tanggal dan witel yang diinputkan. Pada form pencarian ada 2 *button* yaitu *button* cari data dan reset.

8. Menu Detail Data



Gambar 4.16 Menu Detail Data User II

Gambar di atas merupakan menu detail data yang menampilkan data dari *field* tabel yang ada pada menu *list* data. Dimana di dalam submenu detail data ini data yang akan ditampilkan adalah data yang detail serta akan menampilkan *map* (peta) dari data lokasi yang telah diinputkan pada *user* sebelumnya.

9. Menu Laporan

The screenshot shows a web-based application interface for managing data. On the left, a sidebar menu lists 'Inbox', 'Master Mitra', 'List Data Simpro', 'Laporan', 'Optima Data', and 'Optima Request'. The main content area is titled 'Dashboard/Data Simpro' and contains a search bar with a magnifying glass icon and a text input field. Below this is a section for 'List Mitra' with fields for 'Tanggal' (date) and 'Witel' (district), which has a dropdown menu set to 'Bandung'. There are four buttons: 'Cari Data' (Search Data) in green, 'Reset' in blue, 'Print' in blue, and 'Export Excel' and 'Export PDF' in red. Below these buttons is a table titled 'Data Simpro' with 11 columns: Tanggal, ID Simpro, Witel, ID Simpro, Date, STC, UNSC, Alamat, Keterangan, Status, and Aks. Each column has a small arrow icon indicating it is sortable.

Gambar 4.17 Menu Laporan User II

Gambar di atas merupakan menu yang menampilkan data yang telah diinputkan oleh *user* sebelumnya. Yang ditampilkan dalam bentuk tabel. Pada menu ini kita juga bisa melakukan pencarian data berdasarkan tanggal dan witel yang diinputkan. Pada form pencarian ada 2 *button* yaitu *button* cari data dan *reset*.

10. Menu Survey Desain Inventory

Nc	Pekerjaan	STO	Lokasi	Designator	Uraian Pekerjaan	Satuan	Status	Opsi
								<input type="button" value="Edit"/>
								<input type="button" value="Hapus"/>

Gambar 4.18 Menu Survey Desain Inventory User II

Gambar di atas merupakan menu yang menampilkan data yang ada pada menu Optima Data pada submenu SDI. Yang ditampilkan dalam bentuk tabel. Pada menu ini optima bisa melihat data yang masuk dari mitra, kemudian mempunyai hak untuk mengedit dan menghapus data.

11. Menu TA Kontruksi

I	S	Lok	Foto Pemasari	Foto Pemasangai	Foto Pemasan	Foto Pemasang	Stc	Opsi
								<input type="button" value="Edit"/>
								<input type="button" value="Hapus"/>

Gambar 4.19 Menu TA Kontruksi User II

Gambar di atas merupakan menu yang menampilkan data dari menu optima data pada submenu TA Kontruksi. Yang ditampilkan dalam

bentuk tabel. Pada menu ini optima bisa melihat data dari survey yang telah dilakukan oleh mitra.

4.4 Perancangan Arsitektur Perangkat Lunak dan Perangkat Keras Sistem

4.4.1 Perancangan Arsitektur Perangkat Lunak

Tabel 4.5 Arsitektur Perangkat Lunak

No.	<i>Tools/Software</i>	Fungsi	Keterangan
1.	<i>Windows 10</i>	Sistem Operasi	Sistem Operasi
2.	<i>Xampp 3.2.2</i>	Server Basis Data	-
3.	<i>PHP</i>	Pembuatan Sistem Informasi	Bahasa pemrograman
4.	<i>CodeIgniter</i>	<i>Framework</i>	Sebagai pelengkap aplikasi
5.	<i>Visual Studio Code</i>	<i>Text Editor</i>	-

4.4.2 Perancangan Arsitektur Perangkat Keras

Tabel 4.6 Arsitektur Perangkat Keras

No.	Nama Perangkat	Spesifikasi	Keterangan
1.	<i>Processor</i>	<i>Intel core</i>	Untuk kecepatan transfer data dari sistem yang sangat bergantung pada kecepatan prosesor komputer

2.	<i>Memory</i>	1 GB	<i>Memory System</i> yang digunakan
3.	<i>Hardisk</i>	500 GB HDD	Media untuk menyimpan data aplikasi yang dibuat
4.	Infrastruktur jaringan	-	Bisa dianalogikan sebagai alur proses dari titik awal proses sampai pada akhir proses
5.	<i>System Type</i>	64-bit	<i>Operating System</i>

BAB V

TUTORIAL PEMBUATAN SISTEM INFORMASI APPROVAL BERBASIS WEB MENGGUNAKAN FRAMEWORK CODEIGNITER DENGAN NOTIFIKASI E- MAIL

Sebelum memaparkan tutorial dalam pembuatan sistem informasi *approval* ini penulis akan menjelaskan sedikit pemaparan latar belakang mengapa “Sistem Informasi *Approval*” ini dibuat dan hasil yang diperoleh dari aplikasi sudah berhasil dibuat dan diuji. Salah satu permasalahan yang ada di PT. X yaitu bagaimana cara mengalokasikan dan mendokumentasikan permintaan alat produksi dengan baik. Di mana dalam proses pengalokasian dan pendokumentasian tersebut ada beberapa proses salah satunya yaitu adanya persetujuan (*approval*) dari satu *user* ke *user* lainnya.

Berdasarkan uraian di atas maka penulis membuat sebuah “Sistem Informasi *Approval* Berbasis *Web* Menggunakan *Framework CodeIgniter* Dengan Notifikasi *E-mail*”. Bahasa pemrograman yang digunakan adalah *PHP* dengan *framework CodeIgniter* dengan *database MySQL*. Hasil penelitian yang telah penulis lakukan menunjukkan bahwa dengan adanya sistem informasi ini PT. X dapat dengan mudah mengalokasikan serta mendokumentasikan permintaan alat produksi dengan baik.

5.1 Mempersiapkan *Tools*

Untuk membuat sistem informasi *approval* maka diperlukan beberapa *tools* yang dapat mendukung proses pembangunan sistem ini, di mana pada bab sebelumnya telah dijelaskan *tools* apa saja yang digunakan. Berikut beberapa *tools* yang digunakan:

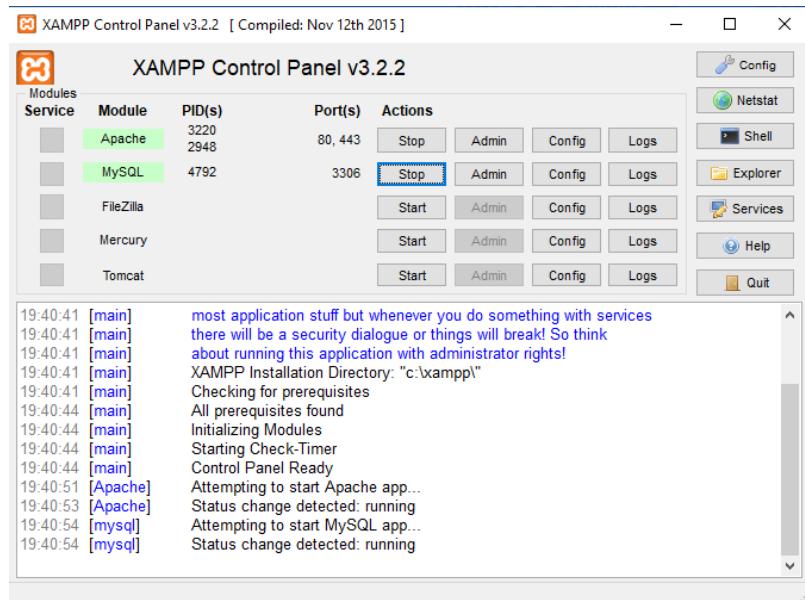
1. XAMPP

2. *Visual Studio Code*

3. *Framework CodeIgniter*

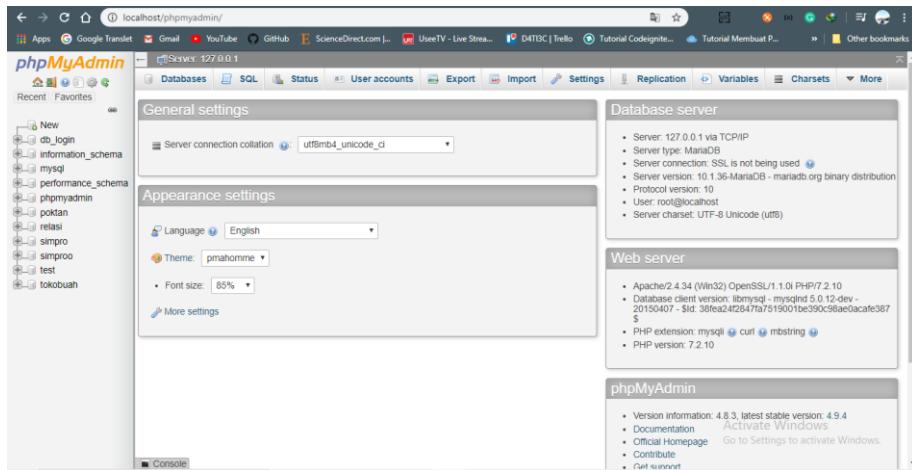
5.2 Membuat Database

1. Hal pertama yang harus dilakukan adalah membuka *XAMPP* di mana *XAMPP* akan memberikan koneksi kepada kita agar bisa mengakses *MySQL* sebagai penyimpanan *database*. Jalankan *XAMPP* dengan klik *start* pada *APACHE* dan *MySQL*.



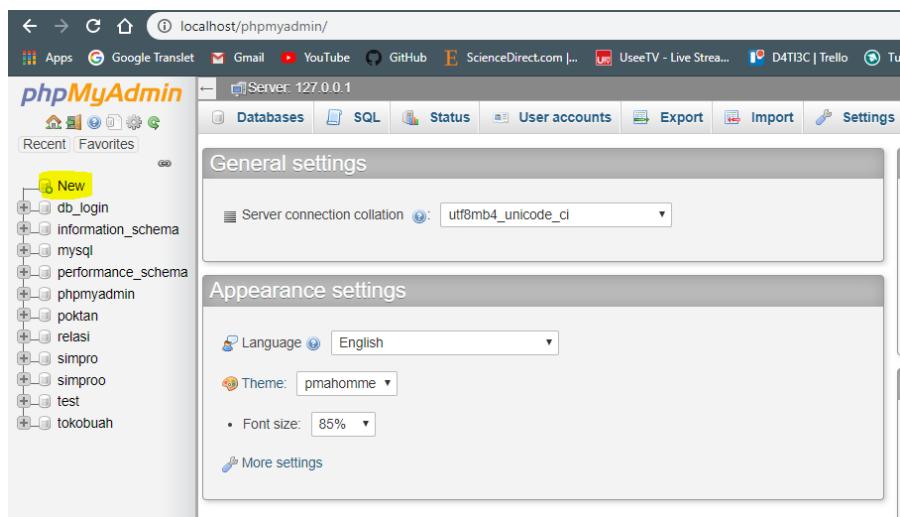
Gambar 5.1 Menjalankan XAMPP

2. Setelah menjalankan aplikasi *XAMPP*, buka *browser* yang ada di perangkat anda, kemudian ketikkan <http://localhost/phpmyadmin/> pada *address bar*.



Gambar 5.2 Localhost

3. Kemudian klik *new* untuk membuat *database* baru.



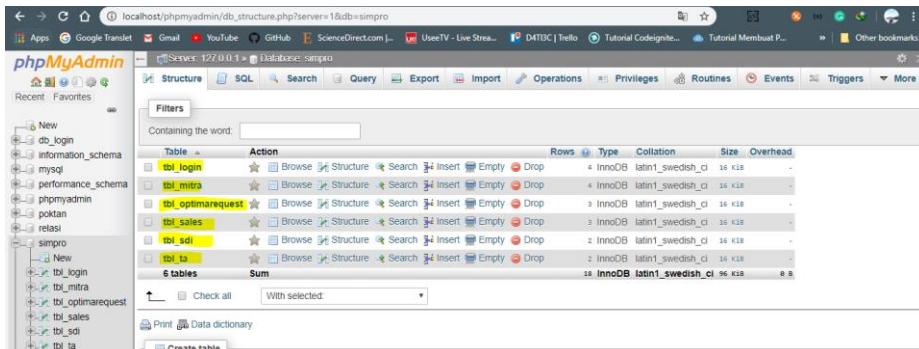
Gambar 5.3 Membuat database baru

4. Pada pembuatan sistem informasi ini saya membuat *database* yang saya butuhkan dengan nama *simpro*.



Gambar 5.4 Membuat database baru

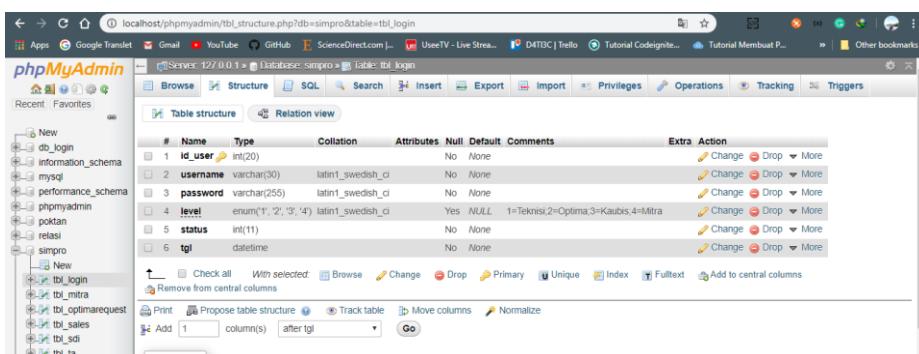
Setelah mengisi nama *database* yang diinginkan selanjutnya klik *create* untuk membuat *database* baru.



Gambar 5.5 Tabel Pada Database Simpro

Pada *database* simpro ini terdapat 6 tabel:

1. Tbl_login



Gambar 5.6 Struktur data *tbl_login*

2. Tbl_mitra

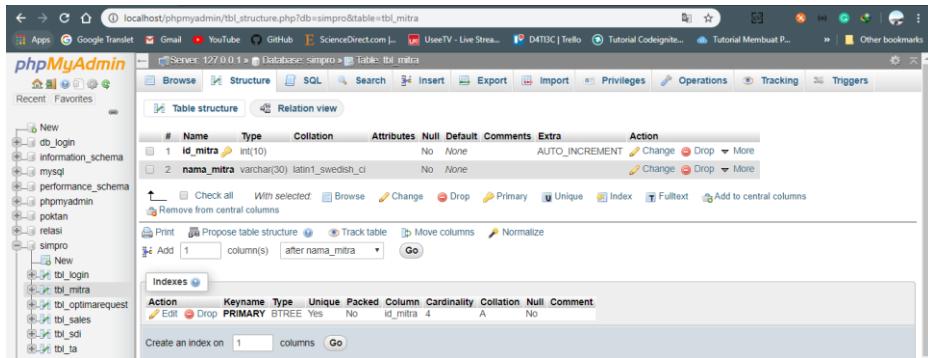


Table structure for Table `tbl_mitra`

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	<code>id_mitra</code>	int(10)			No	None		AUTO_INCREMENT	
2	<code>nama_mitra</code>	varchar(30)	latin1_swedish_ci		No	None			

Indexes

Action	Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
BTREE		BTREE	Yes	No	<code>id_mitra</code>	4	A	No	

Create an index on 1 columns

Gambar 5.7 Struktur data `tbl_mitra`

3. Tbl_optimarequest

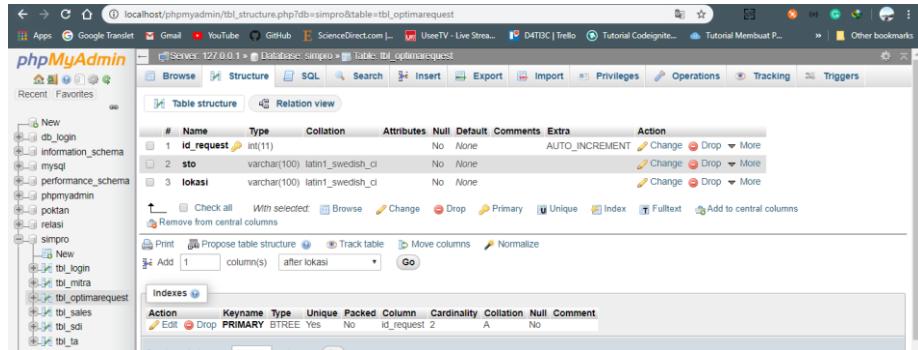


Table structure for Table `tbl_optimarequest`

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	<code>id_request</code>	int(11)			No	None		AUTO_INCREMENT	
2	<code>sto</code>	varchar(100)	latin1_swedish_ci		No	None			
3	<code>lokasi</code>	varchar(100)	latin1_swedish_ci		No	None			

Indexes

Action	Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
BTREE		BTREE	Yes	No	<code>id_request</code>	2	A	No	

Gambar 5.8 Struktur data `tbl_optimarequest`

4. Tbl_sales

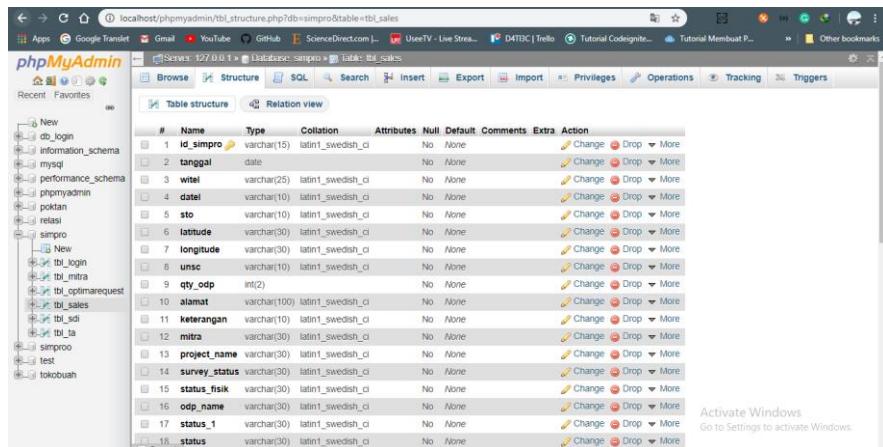


Table structure for Table `tbl_sales`

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	<code>id_simpro</code>	varchar(15)	latin1_swedish_ci		No	None			
2	<code>tangkal</code>	date			No	None			
3	<code>witel</code>	varchar(25)	latin1_swedish_ci		No	None			
4	<code>datel</code>	varchar(10)	latin1_swedish_ci		No	None			
5	<code>sto</code>	varchar(10)	latin1_swedish_ci		No	None			
6	<code>latitude</code>	varchar(30)	latin1_swedish_ci		No	None			
7	<code>longitude</code>	varchar(30)	latin1_swedish_ci		No	None			
8	<code>unsr</code>	varchar(10)	latin1_swedish_ci		No	None			
9	<code>qty_odep</code>	int(2)			No	None			
10	<code>alamat</code>	varchar(100)	latin1_swedish_ci		No	None			
11	<code>keteterangan</code>	varchar(10)	latin1_swedish_ci		No	None			
12	<code>mitra</code>	varchar(30)	latin1_swedish_ci		No	None			
13	<code>project_name</code>	varchar(30)	latin1_swedish_ci		No	None			
14	<code>survey_status</code>	varchar(30)	latin1_swedish_ci		No	None			
15	<code>status_fisik</code>	varchar(30)	latin1_swedish_ci		No	None			
16	<code>odep_name</code>	varchar(30)	latin1_swedish_ci		No	None			
17	<code>status_1</code>	varchar(30)	latin1_swedish_ci		No	None			
18	<code>status</code>	varchar(30)	latin1_swedish_ci		No	None			

Gambar 5.9 Struktur data `tbl_sales`

5. Tbl_sdi

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	id_boq	int(11)			No	None		AUTO_INCREMENT	
2	pekerjaan	varchar(100)	latin1_swedish_ci		No	None			
3	sto	varchar(100)	latin1_swedish_ci		No	None			
4	lokasi	varchar(100)	latin1_swedish_ci		No	None			
5	desimator	enum('DC-OF-SM-4BD','DC-OF-SM-28BD','AC-OF-SM-12','AC-OF-SM-14')	latin1_swedish_ci		No	None			
6	uraian_pekerjaan	enum('Pengadaan dan pemasangan Kabel Duct Fiber Optic')	latin1_swedish_ci		No	None			
7	satuan	enum('meter', 'pc', 'titik')	latin1_swedish_ci		No	None			
8	status	enum('sedang dikerjakan', 'selesai', 'batas', '')	latin1_swedish_ci		No	None			

Gambar 5.10 Struktur data *tbl_sdi*

6. Tbl_ta

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	id_survey	int(11)			No	None		AUTO_INCREMENT	
2	sto	varchar(100)	latin1_swedish_ci		No	None			
3	lokasi	varchar(100)	latin1_swedish_ci		No	None			
4	gambar_oil	varchar(100)	latin1_swedish_ci		No	None			
5	gambar_ftr	varchar(100)	latin1_swedish_ci		No	None			
6	gambar_odc	varchar(100)	latin1_swedish_ci		No	None			
7	gambar_odp	varchar(100)	latin1_swedish_ci		No	None			
8	status	enum('sedang dikerjakan', 'selesai', 'batas', '')	latin1_swedish_ci		No	None			

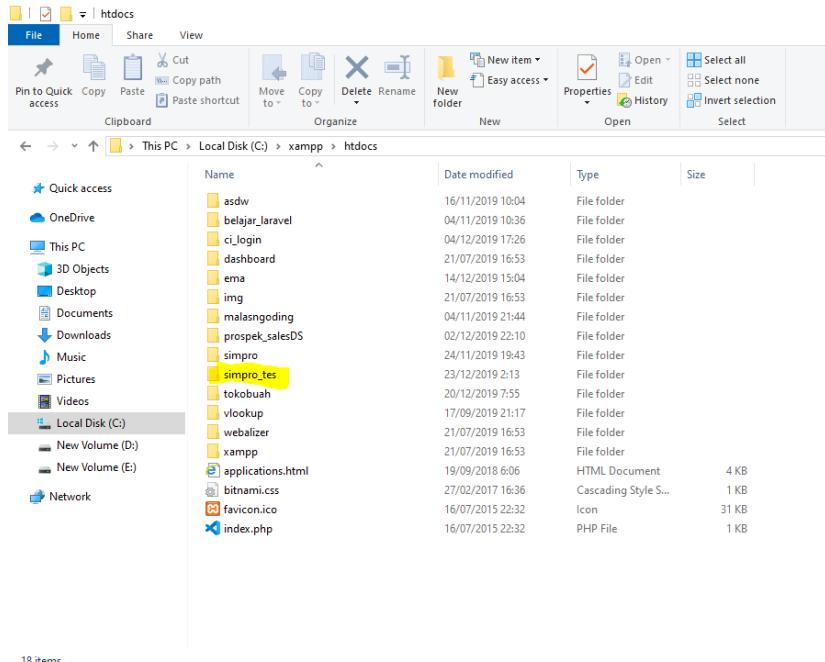
Gambar 5.11 Struktur data *tbl_ta*

Gambar-gambar di atas merupakan struktur *database* yang terdapat pada "Sistem Informasi Approval Berbasis Web Menggunakan *Framework CodeIgniter* Dengan Notifikasi *E-mail*".

5.3 Konfigurasi *CodeIgniter*

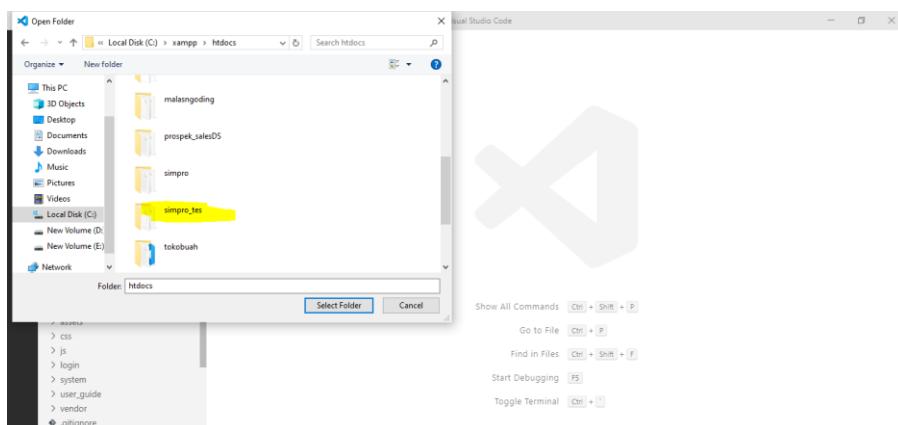
Pada setiap pembuatan aplikasi, sistem infromasi ataupun program harus melakukan konfigurasi terlebih dahulu agar program yang dibuat dapat terhubung dengan *database* yang telah dibuat sebelumnya. Berikut merupakan langkah-langkah dari konfigurasi *CodeIgniter*:

1. Pindahkan folder *CodeIgniter* yang telah *didownload* di <https://codeigniter.com/en/download> kedalam folder C:\xampp\htdocs, kemudia *rename* sesuai dengan kebutuhan anda masing-masing. Pada kasus ini saya memberi nama sistem informasi saya dengan *simpro_tes*.



Gambar 5.12 Folder CodeIgniter Dipindahkan ke htdocs

2. Buka *tools visual studio code*, kemudian buka folder *codeigniter* yang telah di *rename* tadi.



Gambar 5.13 Open Folder

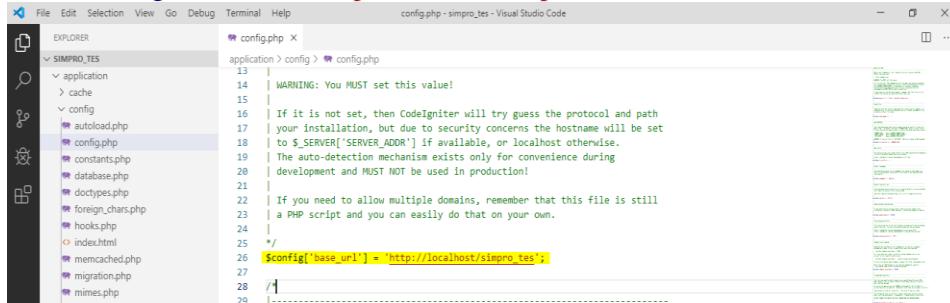
3. Kemudian untuk konfigurasi buka folder *config>config.php*.

Kemudian pada bagian:

`$config['base_url'] =`

isi menjadi:

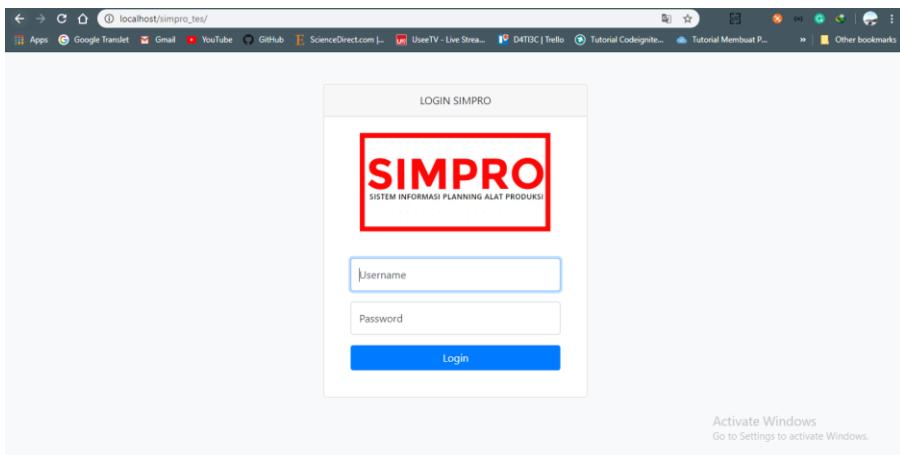
`$config['base_url'] = 'http://localhost/simpro_tes';`



```
File Edit Selection View Go Debug Terminal Help config.php - simpro_tes - Visual Studio Code
EXPLORER config.php X
SIMPRO_TES application > config > config.php
13 | WARNING: You MUST set this value!
14 |
15 |
16 | If it is not set, then CodeIgniter will try guess the protocol and path
17 | your installation, but due to security concerns the hostname will be set
18 | to $_SERVER['SERVER_ADDR'] if available, or localhost otherwise.
19 | The auto-detection mechanism exists only for convenience during
20 | development and MUST NOT be used in production!
21 |
22 | If you need to allow multiple domains, remember that this file is still
23 | a PHP script and you can easily do that on your own.
24 |
25 */
26 $config['base_url'] = 'http://localhost/simpro_tes';
27
28 /
```

Gambar 5.14 Konfigurasi

4. Dan hasilnya jika kita panggil `http://localhost/simpro_tes` pada *browser* maka akan muncul tampilan dari sistem informasi yang kita buat.



Gambar 5.15 Halaman Login Sistem Informasi Approval

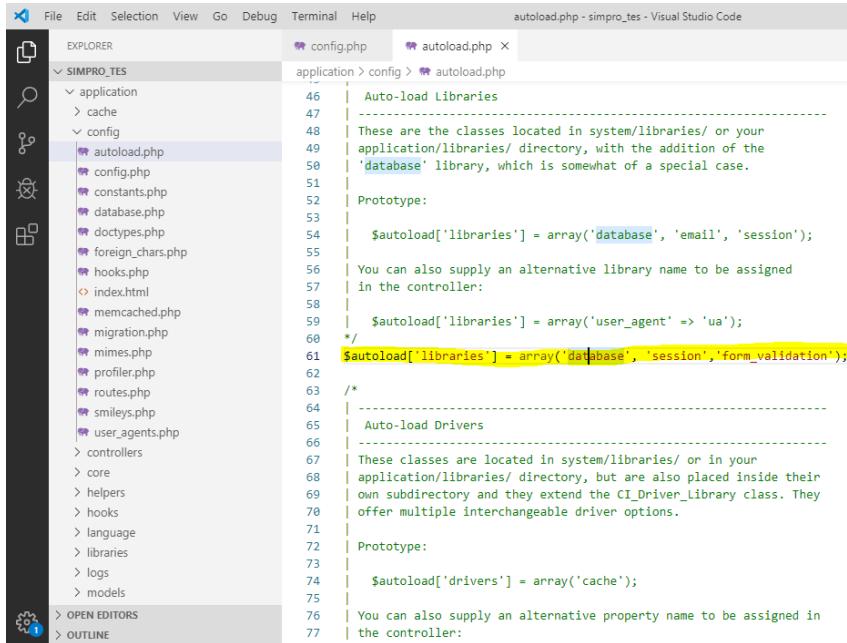
5. Kemudian langkah selanjutnya adalah buka folder *config>autoload.php* pada bagian:

`$autoload['libraries'] =`

isi menjadi:

```
$autoload['libraries'] = array('database', 'session', 'form_validation');
```

Konfigurasi pada file ini bertujuan untuk menentukan sumber daya apa yang akan *diload* secara otomatis nantinya.



```
File Edit Selection View Go Debug Terminal Help
autoload.php - simpro_tes - Visual Studio Code

EXPLORER
SIMPRO_TES
  application
    > cache
    > config
      autoload.php
      config.php
      constants.php
      database.php
      doctypes.php
      foreign_chars.php
      hooks.php
      index.html
      memcached.php
      migration.php
      mimes.php
      profiler.php
      routes.php
      smileys.php
      user_agents.php
    > controllers
    > core
    > helpers
    > hooks
    > language
    > libraries
    > logs
    > models
  > OPEN EDITORS
  > OUTLINE

application > config > autoload.php
46 | Auto-load Libraries
47 | -----
48 | These are the classes located in system/libraries/ or your
49 | application/libraries/ directory, with the addition of the
50 | 'database' library, which is somewhat of a special case.
51 |
52 | Prototype:
53 |
54 | $autoload['libraries'] = array('database', 'email', 'session');
55 |
56 | You can also supply an alternative library name to be assigned
57 | in the controller:
58 |
59 | $autoload['libraries'] = array('user_agent' => 'ua');
60 */
61 $autoload['libraries'] = array('database', 'session', 'form_validation');
62
63 /*
64 | -----
65 | Auto-load Drivers
66 | -----
67 | These classes are located in system/libraries/ or in your
68 | application/libraries/ directory, but are also placed inside their
69 | own subdirectory and they extend the CI_Driver_Library class. They
70 | offer multiple interchangeable driver options.
71 |
72 | Prototype:
73 |
74 | $autoload['drivers'] = array('cache');
75 |
76 | You can also supply an alternative property name to be assigned in
77 | the controller:
```

Gambar 5.16 Konfigurasi Autoload (libraries)

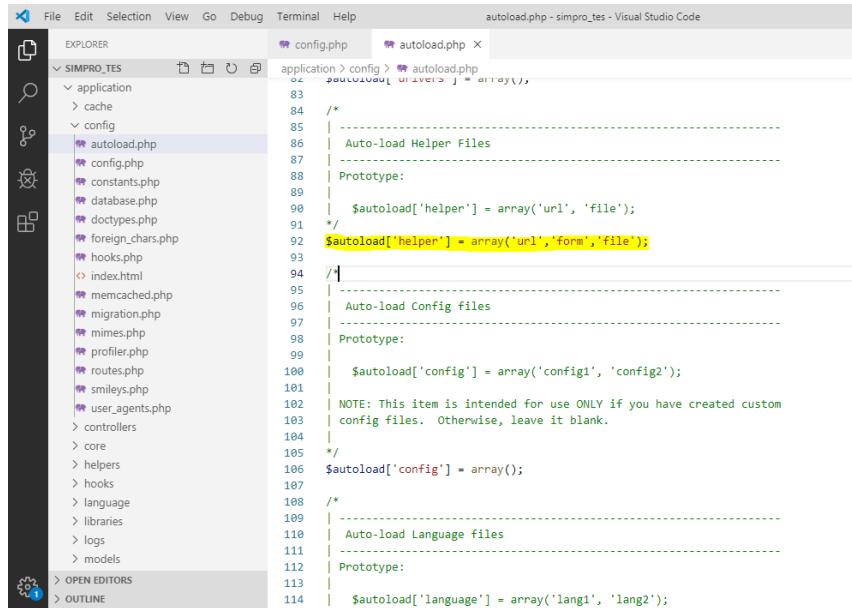
6. Kemudian buka folder *config>autoload.php* pada bagian:

```
$autoload['helper'] =
```

isi menjadi:

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

Konfigurasi pada file ini bertujuan untuk menentukan sumber daya apa yang akan *diload* secara otomatis nantinya.



```

File Edit Selection View Go Debug Terminal Help
autoload.php - simpro_tes - Visual Studio Code
EXPLORER application > config > autoload.php
config.php
autoload.php
constants.php
database.php
doctypes.php
foreign_chars.php
hooks.php
index.html
memcached.php
migration.php
mimes.php
profiler.php
routes.php
smileys.php
user_agents.php
controllers
core
helpers
hooks
language
libraries
logs
models
OPEN EDITORS
OUTLINE
application > config > autoload.php
83
84  /*
85  | -----
86  | Auto-load Helper Files
87  | -----
88  | Prototype:
89  |
90  |     $autoload['helper'] = array('url', 'file');
91  */
92 $autoload['helper'] = array('url', 'form', 'file');
93
94 /**
95 | -----
96 | Auto-load Config files
97 | -----
98 | Prototype:
99
100 |     $autoload['config'] = array('config1', 'config2');
101 |
102 | NOTE: This item is intended for use ONLY if you have created custom
103 | config files. Otherwise, leave it blank.
104 |
105 */
106 $autoload['config'] = array();
107
108 /**
109 | -----
110 | Auto-load Language files
111 | -----
112 | Prototype:
113 |
114 |     $autoload['language'] = array('lang1', 'lang2');

```

Gambar 5.16 Konfigurasi Autoload (helper)

7. Setelah melakukan langkah konfigurasi di atas lanjut pada langkah selanjutnya yaitu pada bagian *config>database.php* kemudian pada bagian:

```

$db['default'] = array(
    'dsn' => '',
    'hostname' => 'localhost',
    'username' => '',
    'password' => '',
    'database' => '',
    'dbdriver' => 'mysqli',

```

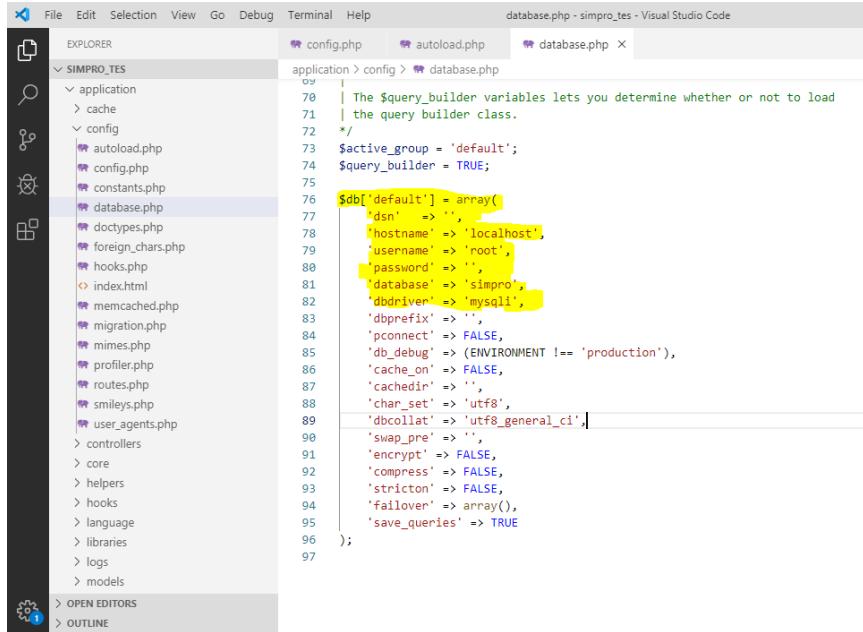
Ubah menjadi:

```

$db['default'] = array(
    'dsn' => '',
    'hostname' => 'localhost',
    'username' => 'root',
    'password' => '',
    'database' => 'simpro',
    'dbdriver' => 'mysqli',

```

Konfigurasi ini berfungsi untuk menghubungkan koneksi *database* yang telah dibuat sebelumnya ke dalam aplikasi *web* yang akan dibuat.



```
File Edit Selection View Go Debug Terminal Help database.php - simpro_tes - Visual Studio Code

EXPLORER application > config > database.php
config
  application > config > database.php
  database.php
  config.php
  constants.php
  autoload.php
  config.php
  database.php
  doctypes.php
  foreign_chars.php
  hooks.php
  index.html
  memcached.php
  migration.php
  mimes.php
  profiler.php
  routes.php
  smileys.php
  user_agents.php
  controllers
  core
  helpers
  hooks
  language
  libraries
  logs
  models

OPEN EDITORS
OUTLINE
```

```
70 | The $query_builder variables lets you determine whether or not to load
71 | the query builder class.
72 */
73 $active_group = 'default';
74 $query_builder = TRUE;
75
76 $db['default'] = array(
77     'dsn' => '',
78     'hostname' => 'localhost',
79     'username' => 'root',
80     'password' => '',
81     'database' => 'simpro',
82     'dbdriver' => 'mysql',
83     'dbprefix' => '',
84     'pconnect' => FALSE,
85     'db_debug' => (ENVIRONMENT !== 'production'),
86     'cache_on' => FALSE,
87     'cacherdir' => '',
88     'char_set' => 'utf8',
89     'dbcollat' => 'utf8_general_ci',
90     'swap_pre' => '',
91     'encrypt' => FALSE,
92     'compress' => FALSE,
93     'stricton' => FALSE,
94     'failover' => array(),
95     'save_queries' => TRUE
96 );
97
```

Gambar 5.17 Konfigurasi Database

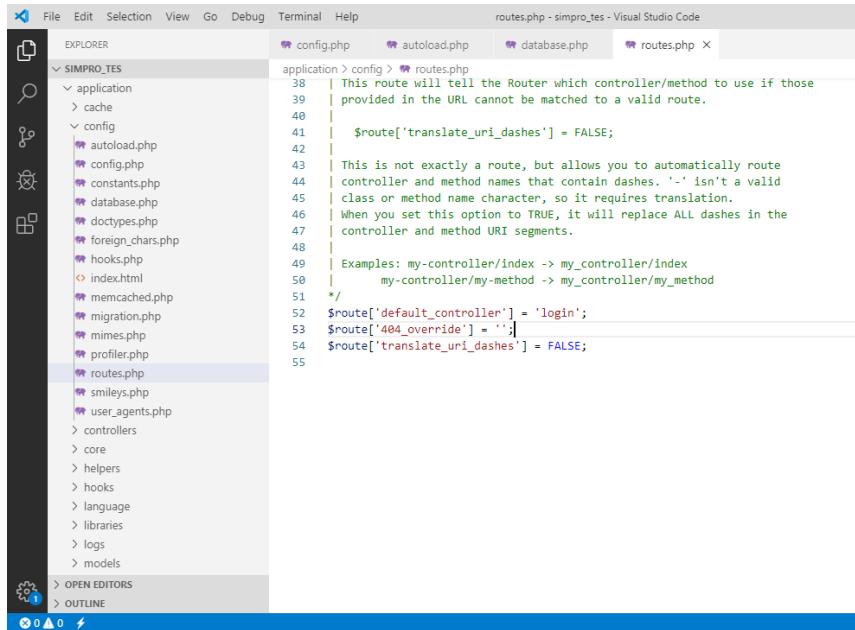
8. Kemudian buka *config>routes.php*, pada bagian:

`$route['default_controller'] = 'welcome';`

Ubah menjadi:

`$route['default_controller'] = 'login';`

Konfigurasi pada file ini bertujuan untuk menentukan *default controller* yang nantinya akan dipanggil.



```
routes.php - simpro_tes - Visual Studio Code
File Edit Selection View Go Debug Terminal Help
routes.php - simpro_tes - Visual Studio Code
EXPLORER routes.php
SIMPRO_TES
  application
    > cache
    > config
      > autoload.php
      > config.php
      > constants.php
      > database.php
      > doctypes.php
      > foreign_chars.php
      > hooks.php
      > index.html
      > memcached.php
      > migration.php
      > mimes.php
      > profiler.php
      > routes.php
      > smileys.php
      > user_agents.php
    > controllers
    > core
    > helpers
    > hooks
    > language
    > libraries
    > logs
    > models
  > OPEN EDITORS
  > OUTLINE
  0 0 0
```

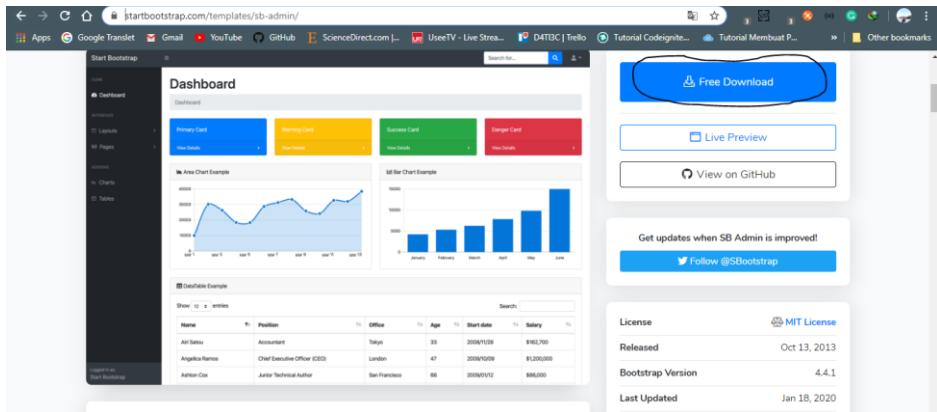
```
application > config > routes.php
38 | This route will tell the Router which controller/method to use if those
39 | provided in the URL cannot be matched to a valid route.
40 |
41 | $route['translate_uri_dashes'] = FALSE;
42 |
43 | This is not exactly a route, but allows you to automatically route
44 | controller and method names that contain dashes. '-' isn't a valid
45 | class or method name character, so it requires translation.
46 | When you set this option to TRUE, it will replace ALL dashes in the
47 | controller and method URI segments.
48 |
49 | Examples: my-controller/index -> my_controller/index
50 |           my-controller/my-method -> my_controller/my_method
51 */
52 $route['default_controller'] = 'login';
53 $route['404_override'] = '';
54 $route['translate_uri_dashes'] = FALSE;
```

Gambar 5.18 Konfigurasi Default Controller

5.4 Konfigurasi *Template CSS Bootstrap*

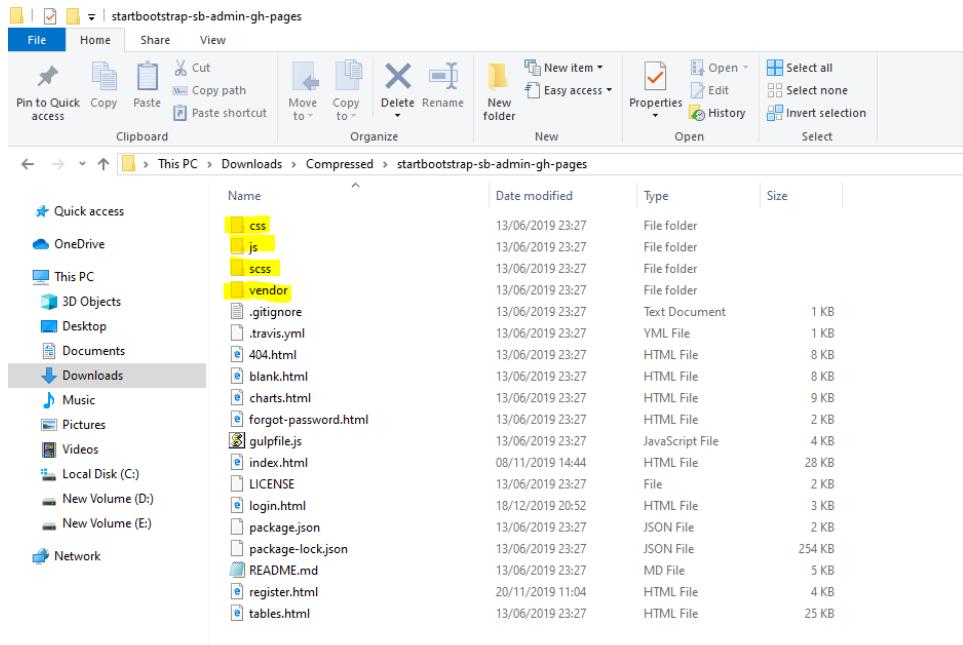
Pada pembuatan sistem informasi, aplikasi ataupun program tentunya harus memiliki tampilan yang menarik. Tujuan dari dibuatnya *template* berfungsi untuk pemanggilan tampilan (*view*) menjadi lebih mudah. Maka dari itu diwajibkan untuk melakukan pengaturan (*setting*) *template* pada *codeigniter* untuk mempermudah pemanggilan *view*. Berikut langkah-langkahnya:

1. Pilih *template* yang anda inginkan atau sesuaikan dengan kebutuhan sistem informasi atau aplikasi yang akan anda bangun. Pada sistem informasi ini saya menggunakan *template bootstrap* dengan nama SB ADMIN. *Template* ini dapat anda unduh di <https://startbootstrap.com/templates/sb-admin/> melalui *web browser* yang ada di perangkat anda.



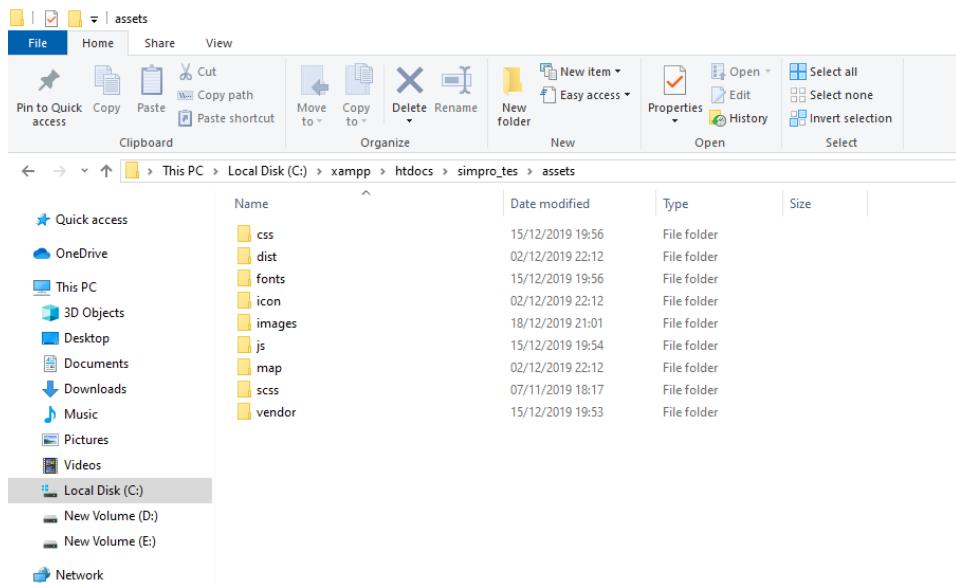
Gambar 5.19 Download Template Bootstrap SB Admin

2. Jika sudah di *download*, *extract* file *bootstrap* tersebut kemudian *copy* beberapa folder seperti *css*, *js*, *scss*, dan *vendor* ke dalam folder *assets* pada folder *simpro_tes* yang ada pada *htdocs*.



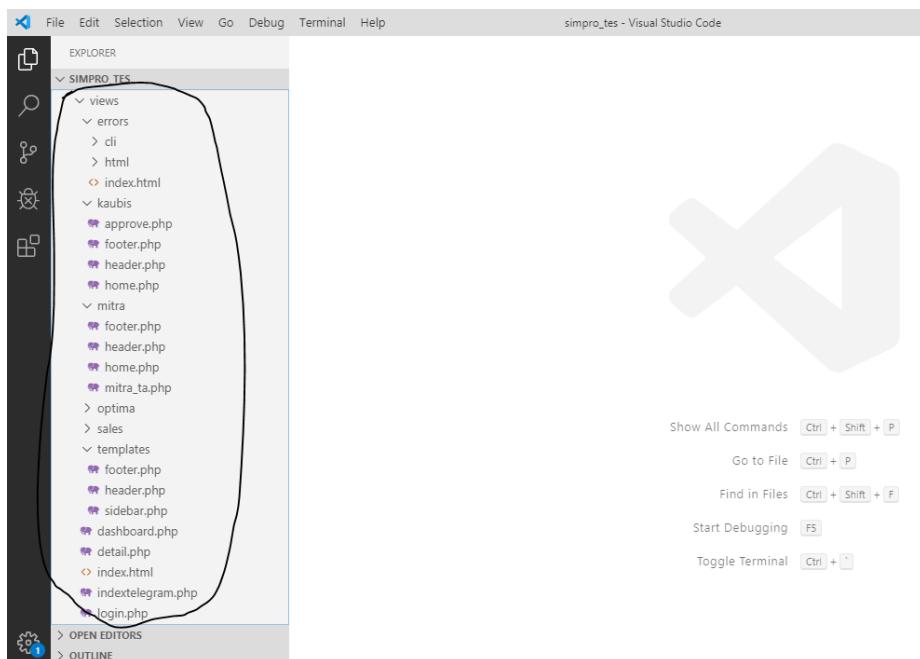
Gambar 5.20 Hasil Extract Folder SB Admin

3. Jika sudah pindahkan file tersebut ke dalam folder *assets* pada folder projek sistem informasi yang akan dibuat.



Gambar 5.21 File Yang Dibutuhkan Untuk Template

4. Langkah selanjutnya adalah pada *visual studio code* buat folder *view* yang dibutuhkan.



Gambar 5.22 Tampilan View

5. Ada beberapa *view* dalam pembuatan sistem infromasi ini:
- Login*
 - Kaubis*
 - Optima*
 - Send Email*

5.5 Tahap Pengkodean (Kodingan)

5.5.1 *Login*

1. *Controller*

login.php
<pre> <?php class Login extends CI_Controller{ function __construct(){ parent::__construct(); \$this->load->model('M_login'); } function index(){ \$this->load->view('login'); } function login() { if (isset(\$_POST['submit'])) { \$username = \$this->input- >post('username',true); \$password = \$this->input- >post('password',true); \$akun = \$this->M_login >cek_akun(\$username,\$password); if(\$akun == 1) { \$this->session- >userdata('username'); \$this->load->view('sales/header'); \$this->load->view('sales/home'); \$this->load->view('sales/footer'); } } } } </pre>

```

        } else if ($akun == 3) {
            $this->session-
>userdata('username');
            $this->load->view('kaubis/footer');
            $this->load->view('kaubis/header');
            $this->load->view('kaubis/home');

        } else if ($akun == 2) {
            $this->session-
>userdata('username');
            $this->load->view('optima/footer');
            $this->load->view('optima/header');
            $this->load->view('optima/home');

        } else if ($akun == 4) {
            $this->session-
>userdata('username');
            $this->load-
>view('templates/header');
            $this->load-
>view('templates/sidebar');
            $this->load->view('dashboard');
            $this->load-
>view('templates/footer');
        }
        else
        {
            echo "<script>alert('Username & Password Salah
')</script>";
            $this->load->view('login');
        }
    }
    else
    {
        echo "<script>alert('BATAL')</script>";
        $this->load->view('login');
    }
}

```

```

        function logout()
        {
    $username    =    $this->session-
>userdata('status_login');
    $this->M_login->logout($username);
    $this->session-
>set_flashdata('pesan', '<div class="alert alert-
success" role="alert"><button type="button" class="
close sucess-op" data-dismiss="alert" aria-
label="Close"> <span class="icon-sc-cl" aria
hidden="true">x</span></button> Logouot Berhasil, Te
rima Kasih </div>');
        redirect(base_url() . "Login/index");
    }
}
}

```

2. *Models*

M_login.php
<pre> <?php class M_login extends CI_Model { function cek_akun(\$username,\$password) { \$query = \$this->db- >get_where('tbl_login',array('username'=>\$username, 'password'=>\$password)); foreach (\$query->result_array() as \$qy) { \$this->session- >set_userdata('username', \$qy['username']); \$this->session- >set_userdata('level', \$qy['level']); } \$row = \$query->row_array(); if (\$query->num_rows()>0) { date_default_timezone_set('Asia/Jakarta'); \$date = date("Y/m/d H:i:s"); } } </pre>

```

        $data    =    array('tgl'=>$date, 'status
'=>1);

        $this->db-
>where('username', $username);
        $this->db->update('tbl_login', $data);
        return $row['level'];
    } else {
        return 0;
    }
}

function logout($username)
{
    $this->db->set(status, 0);
    $this->db->where(username, $username);
    $this->db->update(tbl_login);
}
}

```

3. Views

login.php
<pre> <!DOCTYPE html> <html lang="en"> <head> <meta charset="utf-8"> <meta http-equiv="X-UA Compatible" content="IE=edge"> <meta name="viewport" content="width=device- width, initial-scale=1, shrink-to-fit=no"> <meta name="description" content=""> <meta name="author" content=""> <title>LOGIN</title> <!-- Custom fonts for this template--> <link href="<?php echo base_url('assets/vendor/fo ntawesomefree/css/all.min.css') ?>" rel="styleshe et" type="text/css"> <!-- Page level plugin CSS--> </pre>

```

<link href="<?php echo base_url('assets/vendor/datatables/dataTables.bootstrap4.css') ?>" rel="stylesheet">
<!-- Custom styles for this template-->
<link href="<?php echo base_url('assets/css/admin.css') ?>" rel="stylesheet">
</head>

<body class="bg-light">
<div class="container">
<div class="card card-login mx-auto mt-5">
<div class="card-header" style="text-align: center">LOGIN SIMPRO</div>
<br><center> </center>
<div class="card-body">
<div class="card-body">
<form action="<?= base_url('login/login') ?>" method="POST">
<div class="form-group">
<div class="form-label-group">
<input type="text" name="username" class="form-control" placeholder="Username" required="required" autofocus="autofocus">
<label for="username">Username</label>
</div>
</div>
<div class="form-group">
<div class="form-label-group">
<input type="password" name="password" class="form-control" placeholder="Password" required="required" >
<label for="inputPassword">Password</label>
</div>
</div>
<div class="form-group">
</div>

```

```

        <button class="btn btn-primary btn-block" name="submit">Login</button>
    </form>
    <div class="text-center">

        </div>
    </div>
</div>

<!-- Bootstrap core JavaScript-->
<script src="<?php echo base_url('assets/vendor/jquery/jquery.min.js')?>"></script>
<script src="<?php echo base_url('assets/vendor/bootstrap/js/bootstrap.bundle.min.js')?>"></script>

<!-- Core plugin JavaScript-->
<script src="<?php echo base_url('assets/vendor/jquery-easing/jquery.easing.min.js')?>"></script>

<!-- Page level plugin JavaScript-->
<script src="<?php echo base_url('assets/vendor/chart.js/Chart.min.js')?>"></script>
<script src="<?php echo base_url('assets/vendor/datatables/jquery.dataTables.js')?>"></script>
<script src="<?php echo base_url('assets/vendor/datatables/dataTables.bootstrap4.js')?>"></script>

<!-- Custom scripts for all pages-->
<script src="<?php echo base_url('assets/js/admin.min.js')?>"></script>

<!-- Demo scripts for this page-->
<script src="<?php echo base_url('assets/js/demo/datatables-demo.js')?>"></script>
<script src="<?php echo base_url('assets/js/demo/chart-area-demo.js')?>"></script>
</body>
</html>

```

5.5.2 Kaubis

1. *Controller*

```
kaubis.php

<?php

class kaubis extends CI_Controller{

    function __construct(){
        parent::__construct();
        $this->load->model('M_kaubis');
        $this->load->helper('url');
    }
    public function index()
    {
        $this->load->view('kaubis/header');
        $this->load->view('kaubis/home');
        $this->load->view('kaubis/footer');
    }
    public function approve()
    {
        $data['kaubis']=$this->M_kaubis->show_sales();
        $this->load->view('kaubis/header');
        $this->load->view('kaubis/approve',$data);
    }
    function verifikasi($id_simpro)
    {
        {
            $data['status'] = 'In Progress';
            $this->M_kaubis->up_data($id_simpro,$data);
            redirect('kaubis/approve');
        }
    }
    function declined($id_simpro)
    {
        {
            $data['status'] = 'Declined';
            $this->M_kaubis->tolak($id_simpro,$data);
            redirect('kaubis/approve');
        }
    }
}
```

```
    }
}
```

2. *Models*

```
M_kaubis.php

<?php

class M_kaubis extends CI_Model{
    function show_sales()
    {
        return $this->db->get('tbl_sales')->result();
    }
    function tampil()
    {
        $data = $this->db->get('tbl_sales');
        return $data->result();
    }
    public function input_data($data,$tabel)
    {
        $this->db->insert($tabel,$data);
    }
    function up_data($id_simpro,$data)
    {
        $this->db->where('id_simpro',$id_simpro);
        $this->db->update('tbl_sales',$data);
    }
    function tolak($id_simpro,$data)
    {
        $this->db->where('id_simpro',$id_simpro);
        $this->db->update('tbl_sales',$data);
    }
    public function auto_generate()
    {
// $tgl = date("ymd"); //date(Ymd) : jika mau tahun
4 digit
$this->db-
>select('RIGHT(tbl_sales.id_simpro,3) as kode', FALSE);
$this->db->order_by('id_simpro', 'DESC');
$this->db->limit(1);
$query = $this->db->get('tbl_sales');
if($query->num_rows() <> 0) {
```

```

        $data = $query->row();
        $kode = intval($data->kode) + 1;
    }
    else
    {
        $kode = 1;
    }
    $kodemax = str_pad($kode, 3, 0, STR_PAD_LEFT);
    $kodejadi = "REQ". $kodemax;
    return $kodejadi;
}
}

```

3. Views

home.php
<pre> <body id="page-top"> <nav class="navbar navbar-expand navbar-dark bg-dark static-top"> <a class="navbar-brand mr-1" <?php echo anchor ("kaubis", 'KAUBIS'); ?> <button class="btn btn-link btn-sm text-white order-1 order-sm-0" id="sidebarToggle" href="#"> <i class="fas fa-bars"></i> </button> <!-- Navbar Search --> <form class="d-none d-md-inline-block form-inline ml-auto mr-0 mr-md-3 my-2 my-md-0"> <div class="input-group"> <input type="text" class="form-control" placeholder="Search for..." aria-label="Search" aria-describedby="basic-addon2"> <div class="input-group-append"> <button class="btn btn-primary" type="button"> <i class="fas fa-search"></i> </button> </div> </div> </form> </pre>

```

        </div>
    </div>
</form>

<!-- Navbar -->
<ul class="navbar-nav ml-auto ml-md-0">
    <li class="nav-item dropdown no-arrow mx-1">
        <a class="nav-link dropdown-
toggle" href="#" id="alertsDropdown" role="button"
data-toggle="dropdown" aria-haspopup="true" aria-
expanded="false">
            </a>
        <div class="dropdown-menu dropdown-menu-
right" aria-labelledby="alertsDropdown">
            <a class="dropdown-item" href="#">Action</a>
            <a class="dropdown-
item" href="#">Another action</a>
            <div class="dropdown-divider"></div>
            <a class="dropdown-
item" href="#">Something else here</a>
        </div>
    </li>
    <li class="nav-item dropdown no-arrow">
        <a class="nav-link dropdown-
toggle" href="#" id="userDropdown" role="button" da-
ta-toggle="dropdown" aria-haspopup="true" aria-
expanded="false">
            <i class="fas fa-user-circle fa-fw"></i>
        </a>
        <div class="dropdown-menu dropdown-menu-
right" aria-labelledby="userDropdown">
            <a class="dropdown-item" href="#" data-
toggle="modal" data-
target="#logoutModal">Logout</a>
        </div>
    </li>
</ul>
</nav>
<div id="wrapper">

```

```

<!-- Sidebar -->


- <?php echo anchor ("kaubis/approve", 'Menu Approval',array('class' => 'nav-link')); ?>
</li>

</li>
</ul>
<div id="content-wrapper">
<div class="container-fluid">

<!-- Breadcrumbs-->
<ol class="breadcrumb">
- <?php echo anchor ("kaubis", 'Dashboard'); ?>
</li>
- Kaubis</li>
</ol>

<!-- Icon Cards-->
<p><strong><h3> Selamat Datang Kaubis </strong></p></h3>
<div class="row">
<div class="col-1-3 col-sm-6 mb-3">
<div class="card text-white bg-primary o-hidden h-100">
<div class="card-body">
<div class="card-body-icon">
<i class="fas fa-fw fa-pen"></i>
</div>
<div class="mr-5"> <?php
$koneksi = mysqli_connect("localhost", "root", "", "simpro");
$no = 1;
$data = mysqli_query($koneksi, "select count(*) as hasil from tbl_sales where status='open'");
while($d = mysqli_fetch_array($data)) {

```

```

?>
<tr>
    <td><?php echo $d['hasil']; } ?></td>
    OPEN/NEW</div>
    </div>
    <a class="card-footer text-
white clearfix small z-1" href="#">
        </span>
    </a>
    </div>
</div>
<div class="col-xl-3 col-sm-6 mb-3">
    <div class="card text-white bg-
warning o-hidden h-100">
        <div class="card-body">
            <div class="card-body-icon">
                <i class="fas fa-fw fa-book"></i>
            </div>
            <div class="mr-5"><?php
$koneksi = mysqli_connect("localhost","root","","si
mpro");
$no = 1;
$data = mysqli_query($koneksi,"select count(*)
as hasil from tbl_sales where status='In Progress'"'
);
while($d = mysqli_fetch_array($data)) {
?>
<tr>
    <td><?php echo $d['hasil']; } ?></td> IN PR
OGRESS</div>
    </div>
    <a class="card-footer text-
white clearfix small z-1" href="#">
        </span>
    </a>
    </div>
</div>
<div class="col-xl-3 col-sm-6 mb-3">
    <div class="card text-white bg-
success o-hidden h-100">

```

```

        <div class="card-body">
            <div class="card-body-icon">
                <i class="fas fa-fw fa-
times"></i>
            </div>
            <div class="mr-5"><?php
$koneksi = mysqli_connect("localhost","root","","si
mpro");
$no = 1;
$data = mysqli_query($koneksi,"select count(*)
as hasil from tbl_sales where status='declined'");
while($d = mysqli_fetch_array($data)){
?>
<tr>
    <td><?php echo $d['hasil']; ?></td> DECLI
NED</div>
    </div>
    <a class="card-footer text-
white clearfix small z-1" href="#">
        <span>
        </span>
    </a>
</div>
</div>
<div class="col-1-3 col-sm-6 mb-3">
    <div class="card text-white bg-
danger o-hidden h-100">
        <div class="card-body">
            <div class="card-body-icon">
                <i class="fas fa-fw fa-life-
ring"></i>
            </div>
            <div class="mr-5"><?php
$koneksi = mysqli_connect("localhost","root","","si
mpro");
$no = 1;
$data = mysqli_query($koneksi,"select count(*)
as hasil from tbl_sales where status='close'");
while($d = mysqli_fetch_array($data)){
?>
<tr>

```

```

<td><?php echo $d['hasil']; } ?></td> CLOSE
</div>
</div>
<a class="card-footer text-
white clearfix small z-1" href="#">
    </span>
</a>
</div>
</div>
</div>

</div>
<!-- /.container-fluid -->

<!-- Logout Modal-->
<div class="modal fade" id="logoutModal" tabindex="-1" role="dialog" aria-
labelledby="exampleModalLabel" aria-hidden="true">
    <div class="modal-dialog" role="document">
        <div class="modal-content">
            <div class="modal-header">
                <h5 class="modal-
title" id="exampleModalLabel">Yakin Ingin Meninggal
kan Halaman Ini?</h5>
                <button class="close" type="button" data-
dismiss="modal" aria-label="Close">
                    <span aria-hidden="true">x</span>
                </button>
            </div>
            <div class="modal-
body">Pilih Tombol Logout Jika Kamu ingin Mengingga
ikan Halaman Ini.</div>
            <div class="modal-footer">
                <button class="btn btn-
secondary" type="button" data-
dismiss="modal">Cancel</button>
                <a class="btn btn-
primary" href="<?= base_url(); ?>login/logout" >Log
out</a>
            </div>
        </div>
    </div>
</div>

```

```
    </div>
  </div>
</div>
```

```
approve.php

<!DOCTYPE html>
<html lang="en">

<head>
  <meta charset="utf-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
  <meta name="description" content="">
  <meta name="author" content="">

  <title>KAUBIS</title>
  <!-- Custom fonts for this template-->
  <link href="<?php echo base_url('assets/vendor/font-awesome-free/css/all.min.css') ?>" rel="stylesheet" type="text/css">
  <!-- Page level plugin CSS-->
  <link href="<?php echo base_url('assets/vendor/dataTables/dataTables.bootstrap4.css') ?>" rel="stylesheet">
  <!-- Custom styles for this template-->
  <link href="<?php echo base_url('assets/css/admin.css') ?>" rel="stylesheet">

</head>

<body id="page-top">
<nav class="navbar navbar-expand navbar-dark bg-dark static-top">
<a class="navbar-brand mr-1" <?php echo anchor ("kaubis", 'KAUBIS') ; ?>
```

```

<button class="btn btn-link btn-sm text-
white order-1 order-sm-
0" id="sidebarToggle" href="#">
    <i class="fas fa-bars"></i>
</button>

<!-- Navbar Search -->
<form class="d-none d-md-inline-block form-
inline ml-auto mr-0 mr-md-3 my-2 my-md-0">
    <div class="input-group">
        <input type="text" class="form-
control" placeholder="Search for..." aria-
label="Search" aria-describedby="basic-addon2">
        <div class="input-group-append">
            <button class="btn btn-
primary" type="button">
                <i class="fas fa-search"></i>
            </button>
        </div>
    </div>
</form>

<!-- Navbar -->
<ul class="navbar-nav ml-auto ml-md-0">
    <li class="nav-item dropdown no-arrow mx-1">
        <a class="nav-link dropdown-
toggle" href="#" id="alertsDropdown" role="button" data-
toggle="dropdown" aria-haspopup="true" aria-
expanded="false">
            </a>
        <li class="nav-item dropdown no-arrow">
            <a class="nav-link dropdown-
toggle" href="#" id="userDropdown" role="button" data-
toggle="dropdown" aria-haspopup="true" aria-
expanded="false">
                <i class="fas fa-user-circle fa-fw"></i>
            </a>
            <div class="dropdown-menu dropdown-menu-
right" aria-labelledby="userDropdown">

```

```

        <a class="dropdown-item" href="#" data-
        toggle="modal" data-
        target="#logoutModal">Logout</a>
    </div>
</li>
</ul>
</nav>
<div id="wrapper">

    <!-- Sidebar -->
    <ul class="sidebar navbar-nav">
        <li class="nav-item dropdown">
            <li class="nav-item">
                <?php echo anchor ("kaubis/approve", 'Approval
',array('class' => 'nav-link')); ?>
            </li>
        </ul>
        <div id="content-wrapper">
            <div class="container-fluid">

                <!-- Breadcrumbs-->
                <ol class="breadcrumb">
                    <li class="breadcrumb-item">
                        <?php echo anchor ("kaubis",'Dashboard'); ?>
                    </li>
                    <li class="breadcrumb-item active">Approval</li>
                </ol>
                <p><h7><strong> Anda diminta untuk menyetujui permintaan di bawah ini: </strong></h7>
                    <p>Klik APPROVE untuk menyetujui permintaan , jika tidak setuju klik Non Approve</p>
                    <form class="d-none d-md-inline-block form-
                    inline ml-auto mr-0 mr-md-3 my-2 my-md-0">
                        <div class="input-group">
                            <div class="input-group-append">
                                </button>
                            </div>
                        </div>
                    </form>
                </p>
            </div>
        </div>
    </div>

```

```

<!-- Data -->
<div class="card mb-3">
    <div class="card-header">
        <i class="fas fa-table"></i>
        Data Simpro </div>
    <div class="card-body">
        <div class="table-responsive">
<table class="table table-
bordered" id="dataTable" width="100%" cellspacing="0">
    <thead>
        <tr>
            <th>Tanggal</th>
            <th>#ID Simpro</th>
            <th>Witel</th>
            <th>Datel</th>
            <th>STO</th>
            <th>UNSC</th>
            <th>QTY ODP</th>
            <th>Alamat</th>
            <th>Keterangan</th>
            <th>Status</th>
            <th>Setuju</th>
            <th>Tidak Setuju</th>
        </tr>
    </thead>
    <?php
foreach ($kaubis as $kab) {
    ?>
    <tbody>
        <tr>
            <td><?php echo $kab->tanggal ?></td>
            <td><?php echo $kab->id_simpro ?></td>
            <td><?php echo $kab->witel ?></td>
            <td><?php echo $kab->datel ?></td>
            <td><?php echo $kab->sto ?></td>
            <td><?php echo $kab->unsc ?></td>
            <td><?php echo $kab->qty_odp ?></td>
            <td><?php echo $kab->alamat ?></td>
            <td><?php echo $kab->keterangan ?></td>
        </tr>
    </tbody>

```

```

<td><?php echo $kab->status ?></td>
<td><a href="<?php echo site_url('kaubis/
verifikasiidata/'.$kab-
>id_simpro);?>">Approve</a></td>
<td><a href="<?php echo site_url('kaubis/
declined/'.$kab-
>id_simpro);?>"> Non Approve</a></td>
</td>
</td>
</tr>
<?php } ?>
</tbody>
</table>
</div>
<!-- /.container-fluid -->
<!-- Scroll to Top Button-->
<a class="scroll-to-top rounded" href="#page-
top">
    <i class="fas fa-angle-up"></i>
</a>
<!-- Logout Modal-->
<div class="modal fade" id="logoutModal" tabindex-
=-1" role="dialog" aria-
labelledby="exampleModalLabel" aria-hidden="true">
    <div class="modal-dialog" role="document">
        <div class="modal-content">
            <div class="modal-header">
                <h5 class="modal-
title" id="exampleModalLabel">Ready to Leave?</h5>
                <button class="close" type="button" data-
dismiss="modal" aria-label="Close">
                    <span aria-hidden="true">x</span>
                </button>
            </div>
            <div class="modal-
body">Select "Logout" below if you are ready to end
your current session.</div>
            <div class="modal-footer">

```

```

        <button class="btn btn-
secondary" type="button" data-
dismiss="modal">Cancel</button>
        <a class="btn btn-
primary" <?php echo anchor ("login",'Logout',array(
'class' => 'dropdown-item')); ?></a>
        </div>
    </div>
</div>

<!-- Bootstrap core JavaScript-->
<script src="<?php echo base_url('assets/vendor/jqu
ery/jquery.min.js')?>"></script>
<script src="<?php echo base_url('assets/vendor/boo
tstrap/js/bootstrap.bundle.min.js')?>"></script>

<!-- Core plugin JavaScript-->
<script src="<?php echo base_url('assets/vendor/jqu
ery-easing/jquery.easing.min.js')?>"></script>

<!-- Page level plugin JavaScript-->
<script src="<?php echo base_url('assets/vendor/cha
rt.js/Chart.min.js')?>"></script>
<script src="<?php echo base_url('assets/vendor/dat
atables/jquery.dataTables.js')?>"></script>
<script src="<?php echo base_url('assets/vendor/dat
atables/dataTables.bootstrap4.js')?>"></script>

<!-- Custom scripts for all pages-->
<script src="<?php echo base_url('assets/js/sb-
admin.min.js')?>"></script>

<!-- Demo scripts for this page-->
<script src="<?php echo base_url('assets/js/demo/d
atatables-demo.js')?>"></script>
<script src="<?php echo base_url('assets/js/demo/ch
art-area-demo.js')?>"></script>
</html>
</body>

```

5.5.3 Optima

1. Controller

```
optima.php

<?php

class optima extends CI_Controller{

    function __construct(){
        parent::__construct();

        $this->load->model('M_optima');
        $this->load->model('M_mitra');
        $this->load->helper('url');
    }

    function detail($id_simpro){
        //menampilkan peta dari google maps
        $this->load->library('googlemaps');
        $config['center'] = '-6.916512, 107.611179';
        $config['zoom'] = '15';
        $config['weight'] = '300px';
        $this->googlemaps->initialize($config);
        $pelanggan=$this->M_optima->tampil_data();
        foreach ($pelanggan as $key => $value) {
            $marker = array();
            $marker['animation'] = 'DROP';
            $marker['position'] = "$value->latitude, $value->longitude";
            $marker['infowindow_content'] = '<div class="media" style="widt:300px;">';
            $marker['infowindow_content'] .= '<div class="media-left">';
            $marker['infowindow_content'] .= '</div>';
            $marker['infowindow_content'] .= '<div class="media-body">';
            $marker['infowindow_content'] .= '<h5 class="media-heading">'.$value->witel.'</h5>';
            $marker['infowindow_content'] .= '</div>';
            $marker['infowindow_content'] .= '</div>';
        }
    }
}
```

```

$marker['icon'] = base_url('assets/icon/rumah.png')
;
$this->googlemaps->add_marker($marker);
}
$data['map'] = $this->googlemaps->create_map();
//controller untuk form detail data
$where = array('id_simpro' => $id_simpro);
$data['tbl_sales']=$this->M_optima-
>edit_data($where,'tbl_sales')->result();
$this->load->view('optima/detail',$data);
}
public function index(){
$this->load->view('optima/header');
$this->load->view('optima/home');
$this->load->view('optima/footer');
$data['optima']=$this->M_optima-
>show_sales();
$this->load->view('optima/list', $data);
}
// Untuk menampilkan tampilan list//
public function list(){
$data['optima'] = $this->M_optima-
>tampil_data();
$this->load->view('optima/list', $data);
}
public function create_mitra(){
$this->load->view('optima/header');
$this->load-
>view('optima/create_mitra');
}
function input(){
$this->load->view('list_mitra');
}
public function list_mitra(){
$this->load->view('optima/header');
$data['data'] = $this->M_optima->tampil();
$this->load-
>view('optima/list_mitra', $data);
}
public function inbox() {

```

```

        $data['data']=$this->M_optima-
>show_sales();
        $this->load->view('optima/header');
        $this->load-
>view('optima/inbox',$data  );
    }

    public function data(){
        $this->load->view('optima/header');
        $data['data'] = $this->M_optima-
>tampil_data();
        $this->load-
>view('optima/data', $data);
    }

    public function search(){
        $keyword=$this->input->post('keyword');
        $data['products'] = $this->M_optima-
>get_product_keyword($keyword);
        $this->load-
>view('optima/hasil_data', $data);
    }

    //Untuk input form inputan form pengajuan//
    function input_mitra(){
        $id_mitra = $this->input->post('id_mitra');
        $nama_mitra = $this->input-
>post('nama_mitra');

        $data = array(
            'id_mitra' => $id_mitra,
            'nama_mitra' => $nama_mitra
        );
        $this->M_optima-
>input_data($data,'tbl_mitra');
        redirect('optima/create_mitra');
    }

    function edit($id_mitra){
        $where = array('id_mitra' => $id_mitra);
        $data['data_mitra'] = $this->M_optima-
>edit_data($where,'tbl_mitra')->result();
        $this->load-
>view('optima/update_mitra',$data);
    }
}

```

```

}

function update(){
    $id_mitra = $this->input->post('id');
    $nama_mitra = $this->input-
>post('nama_mitra');

    $data = array(
        'id_mitra' => $id_mitra,
        'nama_mitra' => $nama_mitra
    ) ;

    $where = array(
        'id_mitra' => $id_mitra
    ) ;

    $this->M_optima-
>update_data($where,$data,'tbl_mitra');
    redirect('optima/list_mitra');
}

function hapus($id_mitra){
    $where = array('id_mitra' => $id_mitra);
    $this->M_optima-
>hapus_data($where,'tbl_mitra');
    redirect('optima/list_mitra');
}

//untuk fungsi print inbox
public function print_inbox(){
    $data['optima'] = $this->M_optima-
>tampil_optima("tbl_sales");
    $this->load-
>view('optima/print_inbox', $data);
}

//untuk export pdf inbox
public function pdf_inbox(){
    $this->load->library('dompdf_gen');
    $data['data']=$this->M_optima-
>show_sales();
    $data['optima'] = $this->M_optima-
>tampil_optima("tbl_sales");
}

```

```

        $this->load-
>view('optima/pdf_inbox', $data);

        $paper_size = 'A4';
        $orientation = 'landscape';
        $html = $this->output->get_output();
        $this->dompdf-
>set_paper($paper_size, $orientation);

        $this->dompdf->load_html($html);
        $this->dompdf->render();
        $this->dompdf-
>stream("Inbox Optima.pdf", array('Attachment' =>0)
);
    }
    //untuk export excel inbox optima
    public function excel_inbox(){
        $data['optima'] = $this->M_optima-
>tampil_optima("tbl_sales");

        require(APPPATH. 'PHPExcel-
1.8/Classes/PHPExcel.php');
        require(APPPATH. 'PHPExcel-
1.8/Classes/PHPExcel/Writer/Excel2007.php');

        $object = new PHPExcel();

        $object->getProperties()-
>setCreator("Rahmi Roza");
        $object->getProperties()-
>setLastModifiedBy("Rahmi Roza");
        $object->getProperties()-
>setTitle("Inbox Optima");

        $object->setActiveSheetIndex(0);

        $object->getActiveSheet()-
>setCellValue('A1', '#ID_Simpro');
        $object->getActiveSheet()-
>setCellValue('B1', 'Created');

```

```

        $object->getActiveSheet()-
>setCellValue('C1','QTY ODP');
        $object->getActiveSheet()-
>setCellValue('D1','Alamat');
        $object->getActiveSheet()-
>setCellValue('E1','Keterangan');
        $object->getActiveSheet()-
>setCellValue('F1','Status');

        $baris = 2;
        $no = 1;

        foreach ($data['optima'] as $sal) {
            $object->getActiveSheet()-
>setCellValue('A' . $baris, $sal->id_simpro);
            $object->getActiveSheet()-
>setCellValue('B' . $baris, $sal->created);
            $object->getActiveSheet()-
>setCellValue('C' . $baris, $sal->qty_odp);
            $object->getActiveSheet()-
>setCellValue('D' . $baris, $sal->alamat);
            $object->getActiveSheet()-
>setCellValue('E' . $baris, $sal->keterangan);
            $object->getActiveSheet()-
>setCellValue('F' . $baris, $sal->status);

            $baris++;
        }
        $filename="Inbox Optima".xlsx';

        $object->getActiveSheet()-
>setTitle("Inbox Optima");

        header('Content-
Type: application/vnd.openxmlformats-
officedocument.spreadsheetml.sheet');
        header('Content-
Disposition: attachment;filename="' . $filename. '"';
);
        header('Cache-Control: max-age=0');

```

```

        $writer=PHPExcel_IOFactory::createwriter($object, 'Excel2007');
        $writer->save('php://output');
    }

    //untuk fungsi print list mitra
    public function print_listmitra(){
        $data['optima'] = $this->M_optima->tampil_optimamitra("tbl_mitra");
        $this->load->view('optima/print_listmitra', $data);
    }

    //untuk export pdf list mitra
    public function pdf_listmitra(){
        $this->load->library('dompdf_gen');

        $data['optima'] = $this->M_optima->tampil_optimamitra("tbl_mitra");

        $this->load->view('optima/pdf_listmitra', $data);

        $paper_size = 'A4';
        $orientation = 'landscape';
        $html = $this->output->get_output();
        $this->dompdf->set_paper($paper_size, $orientation);

        $this->dompdf->load_html($html);
        $this->dompdf->render();
        $this->dompdf->stream("List Mitra.pdf", array('Attachment' =>0));
    }

    //untuk export excel list mitra
    public function excel_listmitra(){
        $data['optima'] = $this->M_optima->tampil_optimamitra("tbl_mitra");
    }

```

```

        require(APPPATH. 'PHPEexcel-
1.8/Classes/PHPEexcel.php');
        require(APPPATH. 'PHPEexcel-
1.8/Classes/PHPEexcel/Writer/Excel2007.php');

        $object = new PHPEexcel();

        $object->getProperties()->setCreator("Rahmi Roza");
        $object->getProperties()->setLastModifiedBy("Rahmi Roza");
        $object->getProperties()->setTitle("List Mitra");

        $object->setActiveSheetIndex(0);

        $object->getActiveSheet()->setCellValue('A1', 'No');
        $object->getActiveSheet()->setCellValue('B1', 'Nama Mitra');

        $baris = 2;
        $no = 1;

        foreach ($data['optima'] as $sal) {
            $object->getActiveSheet()->setCellValue('A' . $baris, $sal->id_mitra);
            $object->getActiveSheet()->setCellValue('B' . $baris, $sal->nama_mitra);

            $baris++;
        }
        $filename="List Mitra".'.xlsx';

        $object->getActiveSheet()->setTitle("List Mitra");

        header('Content-
Type: application/vnd.openxmlformats-
officedocument.spreadsheetml.sheet');

```

```

        header('Content-
Disposition: attachment;filename=' . $filename. '""
);
        header('Cache-Control: max-age=0');

        $writer=PHPExcel_IOFactory::createwriter($o
bject, 'Excel2007');
        $writer->save('php://output');
    }

//untuk fungsi print list data simpro optima
public function print_list_data_simpro(){
    $data['optima'] = $this->M_optima-
>tampil_optima("tbl_sales");
    $this->load-
>view('optima/print_listsimpro', $data);
}

//untuk export pdf list data simpro optima
public function pdf_list_data_simpro(){
    $this->load->library('dompdf_gen');

    $data['optima'] = $this->M_optima-
>tampil_optima("tbl_sales");

    $this->load-
>view('optima/pdf_listdatasimpro', $data);

    $paper_size = 'A4';
    $orientation = 'landscape';
    $html = $this->output->get_output();
    $this->dompdf-
>set_paper($paper_size, $orientation);

    $this->dompdf->load_html($html);
    $this->dompdf->render();
    $this->dompdf-
>stream("List Data simpro.pdf", array('Attachment'-
=>0));
}
//untuk export excel list data simpro optima

```

```

public function excel_list_data_simpro() {
    $data['optima'] = $this->M_optima-
>tampil_optima("tbl_sales");

    require(APPPATH. 'PHPEexcel-
1.8/Classes/PHPEexcel.php');
    require(APPPATH. 'PHPEexcel-
1.8/Classes/PHPEexcel/Writer/Excel2007.php');

    $object = new PHPEexcel();

    $object->getProperties()-
>setCreator("Rahmi Roza");
    $object->getProperties()-
>setLastModifiedBy("Rahmi Roza");
    $object->getProperties()-
>setTitle("List Data Simpro Optima");

    $object->setActiveSheetIndex(0);

    $object->getActiveSheet()-
>setCellValue('A1','Tanggal');
    $object->getActiveSheet()-
>setCellValue('B1','#ID Simpro');
    $object->getActiveSheet()-
>setCellValue('C1','Witel');
    $object->getActiveSheet()-
>setCellValue('D1','Datel');
    $object->getActiveSheet()-
>setCellValue('E1','STO');
    $object->getActiveSheet()-
>setCellValue('F1','UNSC');
    $object->getActiveSheet()-
>setCellValue('G1','Qty ODP');
    $object->getActiveSheet()-
>setCellValue('H1','Alamat');
    $object->getActiveSheet()-
>setCellValue('I1','Status');

    $baris = 2;
}

```

```

$no = 1;

foreach ($data['optima'] as $sal) {
    $object->getActiveSheet()-
>setCellValue('A'.'.$baris, $sal->tanggal);
    $object->getActiveSheet()-
>setCellValue('B'.'.$baris, $sal->id_simpro);
    $object->getActiveSheet()-
>setCellValue('C'.'.$baris, $sal->witel);
    $object->getActiveSheet()-
>setCellValue('D'.'.$baris, $sal->dateL);
    $object->getActiveSheet()-
>setCellValue('E'.'.$baris, $sal->sto);
    $object->getActiveSheet()-
>setCellValue('F'.'.$baris, $sal->unsc);
    $object->getActiveSheet()-
>setCellValue('G'.'.$baris, $sal->qty_odp);
    $object->getActiveSheet()-
>setCellValue('H'.'.$baris, $sal->alamat);
    $object->getActiveSheet()-
>setCellValue('I'.'.$baris, $sal->status);

    $baris++;
}
$filename="List Data Simpro Optima'.'.xlsx'
;
$object->getActiveSheet()-
>setTitle("List Data Simpro Optima");

header('Content-
Type: application/vnd.openxmlformats-
officedocument.spreadsheetml.sheet');
header('Content-
Disposition: attachment;filename=' . $filename. ''');
header('Cache-Control: max-age=0');

$writer=PHPExcel_IOFactory::createwriter($o
bject, 'Excel2007');
$writer->save('php://output');

```

```

        }

        //untuk export excel laporan data simpro optima
        public function excel_laporan(){
            $data['optima'] = $this->M_optima->tampil_optima("tbl_sales");

            require(APPPATH. 'PHPEexcel-1.8/Classes/PHPEexcel.php');
            require(APPPATH. 'PHPEexcel-1.8/Classes/PHPEexcel/Writer/Excel2007.php');

            $object = new PHPEexcel();

            $object->getProperties()->setCreator("Rahmi Roza");
            $object->getProperties()->setLastModifiedBy("Rahmi Roza");
            $object->getProperties()->setTitle("Laporan Data Simpro Optima");

            $object->setActiveSheetIndex(0);

            $object->getActiveSheet()->setCellValue('A1','Tanggal');
            $object->getActiveSheet()->setCellValue('B1','#ID Simpro');
            $object->getActiveSheet()->setCellValue('C1','Witel');
            $object->getActiveSheet()->setCellValue('D1','Datel');
            $object->getActiveSheet()->setCellValue('E1','STO');
            $object->getActiveSheet()->setCellValue('F1','UNSC');
            $object->getActiveSheet()->setCellValue('G1','Qty ODP');
            $object->getActiveSheet()->setCellValue('H1','Alamat');
        }
    }
}

```

```

        $object->getActiveSheet()-
>setCellValue('I1','Keterangan');
        $object->getActiveSheet()-
>setCellValue('J1','Mitra');
        $object->getActiveSheet()-
>setCellValue('K1','Project Name');
        $object->getActiveSheet()-
>setCellValue('L1','Survey Status');
        $object->getActiveSheet()-
>setCellValue('M1','Status Fisik');
        $object->getActiveSheet()-
>setCellValue('N1','ODP Name');
        $object->getActiveSheet()-
>setCellValue('O1','Status');
        $object->getActiveSheet()-
>setCellValue('P1','Status');

        $baris = 2;
        $no = 1;

        foreach ($data['optima'] as $sal) {
            $object->getActiveSheet()-
>setCellValue('A' . $baris, $sal->tanggal);
            $object->getActiveSheet()-
>setCellValue('B' . $baris, $sal->id_simpro);
            $object->getActiveSheet()-
>setCellValue('C' . $baris, $sal->witel);
            $object->getActiveSheet()-
>setCellValue('D' . $baris, $sal->dateel);
            $object->getActiveSheet()-
>setCellValue('E' . $baris, $sal->sto);
            $object->getActiveSheet()-
>setCellValue('F' . $baris, $sal->unsc);
            $object->getActiveSheet()-
>setCellValue('G' . $baris, $sal->qty_odp);
            $object->getActiveSheet()-
>setCellValue('H' . $baris, $sal->alamat);
            $object->getActiveSheet()-
>setCellValue('I' . $baris, $sal->keterangan);

```

```

        $object->getActiveSheet()-
>setCellValue('J'.'.$baris, $sal->mitra);
        $object->getActiveSheet()-
>setCellValue('K'.'.$baris, $sal->project_name);
        $object->getActiveSheet()-
>setCellValue('L'.'.$baris, $sal->survey_status);
        $object->getActiveSheet()-
>setCellValue('M'.'.$baris, $sal->status_fisik);
        $object->getActiveSheet()-
>setCellValue('N'.'.$baris, $sal->odp_name);
        $object->getActiveSheet()-
>setCellValue('O'.'.$baris, $sal->status_1);
        $object->getActiveSheet()-
>setCellValue('P'.'.$baris, $sal->status);

        $baris++;
    }
    $filename="Laporan Data Simpro Optima".
'.xlsx';

        $object->getActiveSheet()-
>setTitle("Laporan Data Simpro Optima");

        header('Content-
Type: application/vnd.openxmlformats-
officedocument.spreadsheetml.sheet');
        header('Content-
Disposition: attachment;filename=' . $filename. ''');
);
        header('Cache-Control: max-age=0');

        $writer=PHPExcel_IOFactory::createWriter
($object, 'Excel2007');
        $writer->save('php://output');
    }
}

```

2. *Model*

optima.php

<?php class M_optima extends CI_Model{
--

```

        // function show_sales1(){
        // $hasil=$this->db-
>query("SELECT * FROM tbl_sales");
        // return $hasil;
        // }
        function show_sales()
        {
        $this->db-
>select('id_simpro,tanggal,created,qty_odp,alamat,k
eterangan,status,witel,datel,sto,unsc,(datediff(NOW
(),tanggal)) as durasi');
        $this->db->from('tbl_sales');
        return $this->db->get()->result();
        }
        function tampil_data()
        {
        $data = $this->db->get('tbl_sales');
        return $data->result();
        }
        function tampil_optima()
        {
        $data = $this->db->get('tbl_sales');
        return $data->result();
        }
        function tampil_optimamitra()
        {
        $data = $this->db->get('tbl_mitra');
        return $data->result();
        }
        function tampil()
        {
        $data = $this->db->get('tbl_mitra');
        return $data->result();
        }
        public function input_data($data,$tabel)
        {
        $this->db->insert($tabel,$data);
        }
        public function durasi(){

```

```

    $this->db-
>select('datediff(NOW(),tanggal) as durasi');
    $this->db->from('tbl_sales');
    return $this->db->get()->result();
}
public function auto_generate()
{
    // $tgl = date("ymd"); //date(Ymd) : jika mau tahun 4 digit
    $this->db-
>select('RIGHT(tbl_sales.id_simpro,3) as kode', FALSE);
    $this->db->order_by('id_simpro', 'DESC');
    $this->db->limit(1);
    $query = $this->db->get('tbl_sales');
    if($query->num_rows() <> 0) {
        $data = $query->row();
        $kode = intval($data->kode) + 1;
    }
    else
    {
        $kode = 1;
    }
    $kodemax = str_pad($kode, 3, 0, STR_PAD_LEFT);
    $kodejadi = "REQ". $kodemax;
    return $kodejadi;
}
public function get_product_keyword($keyword) {
    $this->db->select('*');
    $this->db->from('tbl_sales');
    $this->db->like('tanggal',$keyword);
    return $this->db->get()->result();
}
function edit_data($where,$table){
    return $this->db->get_where($table,$where);
}
function update_data($where,$data,$table) {
    $this->db->where($where);
    $this->db->update($table,$data);
}

```

```

function hapus_data($where, $table) {
    $this->db->where($where);
    $this->db->delete($table);
}
}

```

3. Views

inbox.php
<pre> <!DOCTYPE html> <html lang="en"> <head> <meta charset="utf-8"> <meta http-equiv="X-UA-Compatible" content="IE=edge"> <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no"> <meta name="description" content=""> <meta name="author" content=""> <title>KAUBIS</title> <!-- Custom fonts for this template--> <link href="<?php echo base_url('assets/vendor/fontawesome-free/css/all.min.css') ?>" rel="stylesheet" type="text/css"> <!-- Page level plugin CSS--> <link href="<?php echo base_url('assets/vendor/datatables/dataTables.bootstrap4.css') ?>" rel="stylesheet"> <!-- Custom styles for this template--> <link href="<?php echo base_url('assets/css/sb-admin.css') ?>" rel="stylesheet"> </head> <body id="page-top"> <nav class="navbar navbar-expand navbar-dark bg-dark static-top"> </pre>

```

<a class="navbar-brand mr-1" <?php echo anchor ("optima", 'Optima'); ?>

<button class="btn btn-link btn-sm text-white order-1 order-sm-0" id="sidebarToggle" href="#"></button>

<!-- Navbar Search -->
<form class="d-none d-md-inline-block form-inline ml-auto mr-0 mr-md-3 my-2 my-md-0">
<div class="input-group">
    <input type="text" class="form-control" placeholder="Search for..." aria-label="Search" aria-describedby="basic-addon2">
        <div class="input-group-append">
            <button class="btn btn-primary" type="button">
                <i class="fas fa-search"></i>
            </button>
        </div>
    </div>
</form>

<!-- Navbar -->
<ul class="navbar-nav ml-auto ml-md-0">
    <li class="nav-item dropdown no-arrow mx-1">
        <a class="nav-link dropdown-toggle" href="#" id="alertsDropdown" role="button" data-toggle="dropdown" aria-haspopup="true" aria-expanded="false">
            </a>
        <li class="nav-item dropdown no-arrow">
            <a class="nav-link dropdown-toggle" href="#" id="userDropdown" role="button" data-toggle="dropdown" aria-haspopup="true" aria-expanded="false">
                <i class="fas fa-user-circle fa-fw"></i>
            </a>
        </li>
    </li>
</ul>

```

```

        <div class="dropdown-menu dropdown-menu-right" aria-labelledby="userDropdown">
            <a class="dropdown-item" href="#" data-toggle="modal" data-target="#logoutModal">Logout</a>
        </div>
    </li>
</ul>
</nav>

<div id="wrapper">

<!-- Sidebar -->
<ul class="sidebar navbar-nav">
    <li class="nav-item dropdown">
        <li class="nav-item">
<?php echo anchor ("optima/inbox", 'Inbox', array('class' => 'nav-link')); ?>
        </li>
        <li class="nav-item active">
        <li class="nav-item dropdown">
            <a class="nav-link dropdown-toggle" href="#" id="pagesDropdown" role="button" data-toggle="dropdown" aria-haspopup="true" aria-expanded="false">
                <i class="fas fa-fw fa-database"></i>
                <span>Master Mitra</span>
            </a>
            <div class="dropdown-menu" aria-labelledby="pagesDropdown">
                <?php echo anchor ("optima/create_mitra", 'Create Mitra', array('class' => 'dropdown-item')); ?>
                <?php echo anchor ("optima/list_mitra", 'List Mitra', array('class' => 'dropdown-item')); ?>
            </div>
            <li class="nav-item">
                <?php echo anchor ("optima/list", 'List Data Simpro', array('class' => 'nav-link')); ?>
            <li class="nav-item dropdown">

```

```

        <a class="nav-link dropdown-
toggle" href="#" id="pagesDropdown" role="button" d
ata-toggle="dropdown" aria-haspopup="true" aria-
expanded="false">
            <i class="fas fa-fw fa-folder"></i>
            <span>Laporan</span>
        </a>
        <div class="dropdown-menu" aria-
labelledby="pagesDropdown">
            <?php echo anchor ("optima/data",'Data Si
mpro',array('class' => 'dropdown-item')); ?>
        </div>
    </li>
    <li class="nav-item dropdown">
        <a class="nav-link dropdown-
toggle" href="#" id="pagesDropdown" role="button" d
ata-toggle="dropdown" aria-haspopup="true" aria-
expanded="false">
            <i class="fas fa-fw fa-book"></i>
            <span>Optima Data</span>
        </a>
        <div class="dropdown-menu" aria-
labelledby="pagesDropdown">
            <?php echo anchor ("optimasdi",'Survey De
sain Inventory (SDI)',array('class' => 'dropdown-
item')); ?>
            <?php echo anchor ("optimata",'TA Kontruk
si',array('class' => 'dropdown-item')); ?>
        </div>
    </li>
</ul>
<div id="content-wrapper">
    <div class="container-fluid">

<!-- Breadcrumbs-->
    <ol class="breadcrumb">
        <li class="breadcrumb-item">
            <?php echo anchor ("optima",'Dashboard'); ?>
        </li>
    </ol>
</div>

```

```

        <li class="breadcrumb-item active">Inbox</li>
    </ol>
    <div class="container-fluid cm-container-white">
        <h2 style="margin-top:0;"><i class="fa fa-inbox"></i> Inbox </h2>
        </div>
        <div>
            <a class="btn btn-info" href="< ?php echo base_url('optima/print_inbox') ?>"> <i class="fa fa-print"></i> Print</a>
            <a class="btn btn-danger" href="< ?php echo base_url('optima/pdf_inbox') ?>"> <i class="fa fa-file"></i> Export PDF</a>
            <a class="btn btn-success" href="< ?php echo base_url('optima/excel_inbox') ?>"> <i class="fa fa-download"></i> Export Excel</a>
        </div>
        <form class="d-none d-md-inline-block form-inline ml-auto mr-0 mr-md-3 my-2 my-md-0">
            <div class="input-group">
                <div class="input-group-append">
                    <button>
                </div>
            </div>
        </form>

        <!-- Data -->
        <div class="card mb-3">
            <div class="card-header">
                <i class="fas fa-inbox"></i>
                Data Simpro </div>
            <div class="card-body">
                <div class="table-responsive">
                    <table class="table table-bordered" id="dataTable" width="100%" cellspacing="0">

```

```

<thead>
    <tr>
        <th>#ID Simpro</th>
        <th>Time Durasi</th>
        <th>Created</th>
        <th>QTY ODP</th>
        <th>Alamat</th>
        <th>Keterangan</th>
        <th>Status</th>
        <th>Kirim Pesan</th>
    </tr>
</thead>
<?php
foreach($data as $sal) {
?>
<tbody>
<tr>
    <td><?php echo anchor('optima/detail/.'.$sal->id_simpro, 'ID Simpro'); ?></td>
    <td><?php echo $sal->durasi ?> hari</td>
    <td><?php echo $sal->created ?></td>
    <td><?php echo $sal->qty_odp ?></td>
    <td><?php echo $sal->alamat ?></td>
    <td><?php echo $sal->keterangan ?></td>
    <td><?php echo $sal->status ?></td>
    <td><a class="btn btn-danger" href="<?php echo base_url().'sendmail_message';?>"><i class="fa fa-fa-print"></i>Kirim</a></td>
</tr>
<?php } ?>
</tbody>
</table>
</div>
<!-- /.container-fluid -->

```

```

<!-- Scroll to Top Button-->
<a class="scroll-to-top rounded" href="#page-top">
    <i class="fas fa-angle-up"></i>
</a>

<!-- Logout Modal-->
<div class="modal fade" id="logoutModal" tabindex="-1" role="dialog" aria-labelledby="exampleModalLabel" aria-hidden="true">
    <div class="modal-dialog" role="document">
        <div class="modal-content">
            <div class="modal-header">
                <h5 class="modal-title" id="exampleModalLabel">Ready to Leave?</h5>
                <button class="close" type="button" data-dismiss="modal" aria-label="Close">
                    <span aria-hidden="true">×</span>
                </button>
            </div>
            <div class="modal-body">Select "Logout" below if you are ready to end your current session.</div>
            <div class="modal-footer">
                <button class="btn btn-secondary" type="button" data-dismiss="modal">Cancel</button>
                <a class="btn btn-primary" href="<?php echo anchor ('login','Logout',array('class' => 'dropdown-item')); ?>"></a>
            </div>
        </div>
    </div>
</div>

<!-- Bootstrap core JavaScript-->
<script src="<?php echo base_url('assets/vendor/jquery/jquery.min.js')?>"></script>
<script src="<?php echo base_url('assets/vendor/bootstrap/js/bootstrap.bundle.min.js')?>"></script>

```

```

<!-- Core plugin JavaScript-->
<script src="<?php echo base_url('assets/vendor/jquery-easing/jquery.easing.min.js')?>"></script>
<!-- Page level plugin JavaScript-->
<script src="<?php echo base_url('assets/vendor/chart.js/Chart.min.js')?>"></script>
<script src="<?php echo base_url('assets/vendor/datatables/jquery.dataTables.js')?>"></script>
<script src="<?php echo base_url('assets/vendor/datatables/dataTables.bootstrap4.js')?>"></script>
<!-- Custom scripts for all pages-->
<script src="<?php echo base_url('assets/js/admin.min.js')?>"></script>
<!-- Demo scripts for this page-->
<script src="<?php echo base_url('assets/js/demo/datatables-demo.js')?>"></script>
<script src="<?php echo base_url('assets/js/demo/chart-area-demo.js')?>"></script>

</html>
</body>

```

5.5.4 Send Mail

1. *Controller*

Sendmail_message.php
<pre> <?php defined('BASEPATH') OR exit('No direct script access allowed'); /** * */ class Sendmail_message extends CI_Controller { function index() { \$this->load->view('optima/header'); \$this->load->view('sendmail_message'); \$this->load->view('optima/footer'); } } </pre>

```

function send(){
    $email      = $this->input-
>post('email_address');
    $psn       = $this->input->post('message_text');
    $this->form_validation-
>set_rules('email_address','Email Address','require
d');
    $this->form_validation-
>set_rules('message_text','Message Text','required'
);

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

        $pesan = '<center><font color="green"><strong
>----- PEMBERITAHUAN -----
- </strong></font></center>
<center><font color="black"><br><br> Ke
pada Yth. Bapak/Ibu : '.$email.
        '<br>Pesan : '.$psn.
        '</center><br><br></font><center><font
color="green"><strong>-----
- PT. TELKOM INDONESIA -----
- </strong></font></center>';
        // Konfigurasi email

        $config1 = [
            'mailtype'  => 'html',
            'charset'   => 'utf-8',
            'protocol'  => 'smtp',
            'smtp_host' => 'smtp.gmail.com',
            'smtp_user' => 'internship1wildan@gma
il.com',// Email gmail
            'smtp_pass'  => 'npm1164058', // Password
gmail
            'smtp_crypto' => 'ssl',
            'smtp_port'   => 465,
            'crlf'        => "\r\n",
            'newline'     => "\r\n"
        ];
    }
}

```

```

// Load library email dan konfigurasinya
$this->load->library('email', $config1);
// Email dan nama pengirim
$this->email
>from('internship1wildan@gmail.com', 'PT. Tel
kom (Indonesia)');
// Email penerima
$this->email-
>to($email); // Ganti dengan email tujuan
// Subject email
$this->email-
>subject('Pemberitahuan - PT. Telkom Indonesia
a (No-Reply)');
// Isi email
$this->email->message($pesan);
// Tampilkan pesan sukses atau error
if ($this->email->send()) {
    echo 'Sukses! email berhasil dikirim.';
} else {
    echo 'Error! email tidak dapat dikirim.';
}
$this->session-
>set_flashdata('flash', 'Berhasil ditambahkan
');
redirect(base_url().'optima/inbox');
} else {
$this->load->view('optima/header');
$this->load->view('sendmail_message');
$this->load->view('optima/footer');
}
}
?>

```

2. View

Sendmail_message.php
<!DOCTYPE html> <html lang="en"> <head> <meta charset="utf-8">

```

<meta http-equiv="X-UA-Compatible" content="IE=edge">
<meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
<meta name="description" content="">
<meta name="author" content="">

<title>OPTIMA</title>
<!-- Custom fonts for this template-->
<link href="<?php echo base_url('assets/vendor/font-awesome-free/css/all.min.css') ?>" rel="stylesheet" type="text/css">
<!-- Page level plugin CSS-->
<link href="<?php echo base_url('assets/vendor/datatables/dataTables.bootstrap4.css') ?>" rel="stylesheet">
<!-- Custom styles for this template-->
<link href="<?php echo base_url('assets/css/admin.css') ?>" rel="stylesheet">
</head>

<body id="page-top">
<nav class="navbar navbar-expand navbar-dark bg-dark static-top">

<a class="navbar-brand mr-1" <?php echo anchor ("optima", 'Optima'); ?>
<button class="btn btn-link btn-sm text-white order-1 order-sm-0" id="sidebarToggle" href="#">

```

```

<input type="text" class="form-control" placeholder="Search for..." aria-label="Search" aria-describedby="basic-addon2">
<div class="input-group-append">
    <button class="btn btn-primary" type="button">
        <i class="fas fa-search"></i>
    </button>
</div>
</div>
</form>

<!-- Navbar -->
<ul class="navbar-nav ml-auto ml-md-0">
    <li class="nav-item dropdown no-arrow mx-1">
        <a class="nav-link dropdown-toggle" href="#" id="alertsDropdown" role="button" data-toggle="dropdown" aria-haspopup="true" aria-expanded="false">
            </a>
        <li class="nav-item dropdown no-arrow">
            <a class="nav-link dropdown-toggle" href="#" id="userDropdown" role="button" data-toggle="dropdown" aria-haspopup="true" aria-expanded="false">
                <i class="fas fa-user-circle fa-fw"></i>
            </a>
            <div class="dropdown-menu dropdown-menu-right" aria-labelledby="userDropdown">
                <a class="dropdown-item" href="#" data-toggle="modal" data-target="#logoutModal">Logout</a>
            </div>
        </li>
    </ul>
</nav>
<div id="wrapper">

<!-- Sidebar -->
<ul class="sidebar navbar-nav">
    <li class="nav-item dropdown">

```

```

<li class="nav-item">
<?php echo anchor ("optima/inbox", 'Inbox',array('class' => 'nav-link')); ?>
</li>
<li class="nav-item active">
<li class="nav-item dropdown">
<a class="nav-link dropdown-toggle" href="#" id="pagesDropdown" role="button" data-toggle="dropdown" aria-haspopup="true" aria-expanded="false">
<i class="fas fa-fw fa-database"></i>
<span>Master Mitra</span>
</a>
<div class="dropdown-menu" aria-labelledby="pagesDropdown">
<?php echo anchor ("optima/create_mitra", 'Create Mitra',array('class' => 'dropdown-item')); ?>
<?php echo anchor ("optima/list_mitra", 'List Mitra',array('class' => 'dropdown-item')); ?>
</div>
<li class="nav-item">
<?php echo anchor ("optima/list", 'List Data Simpro',array('class' => 'nav-link')); ?>
<li class="nav-item dropdown">
<a class="nav-link dropdown-toggle" href="#" id="pagesDropdown" role="button" data-toggle="dropdown" aria-haspopup="true" aria-expanded="false">
<i class="fas fa-fw fa-folder"></i>
<span>Laporan</span>
</a>
<div class="dropdown-menu" aria-labelledby="pagesDropdown">
<?php echo anchor ("optima/data", 'Data Simpro',array('class' => 'dropdown-item')); ?>
</div>
</li>
<li class="nav-item dropdown">

```

```

        <a class="nav-link dropdown-
toggle" href="#" id="pagesDropdown" role="button" d
ata-toggle="dropdown" aria-haspopup="true" aria-
expanded="false">
            <i class="fas fa-fw fa-book"></i>
            <span>Optima Data</span>
        </a>
        <div class="dropdown-menu" aria-
labelledby="pagesDropdown">
            <?php echo anchor ("optima/sdi", 'Survey D
esain Inventory (SDI)',array('class' => 'dropdown-
item')); ?>
            <?php echo anchor ("optima/data", 'TA Kont
ruksi',array('class' => 'dropdown-item')); ?>
        </div>
    </li>
    <li class="nav-item">
        <?php echo anchor ("optima/optimarequest", 'Op
tima Request',array('class' => 'nav-link')); ?>
    </li>
</ul>
<div id="content-wrapper">
    <div class="container-fluid">

<!-- Breadcrumbs-->
<ol class="breadcrumb">
    <li class="breadcrumb-item">
        <?php echo anchor ("optima", 'Dashboard'); ?>
    </li>
    <li class="breadcrumb-
item active"> Kirim Pesan Email </li>
</ol>
    <div class="col-md-4">
        <div class="box box-solid box-primary">
            <div class="box-header">
<h4 class="box-
title"><b>Email Message <i class="fa fa-
send"></i></b></h4>
            </div>

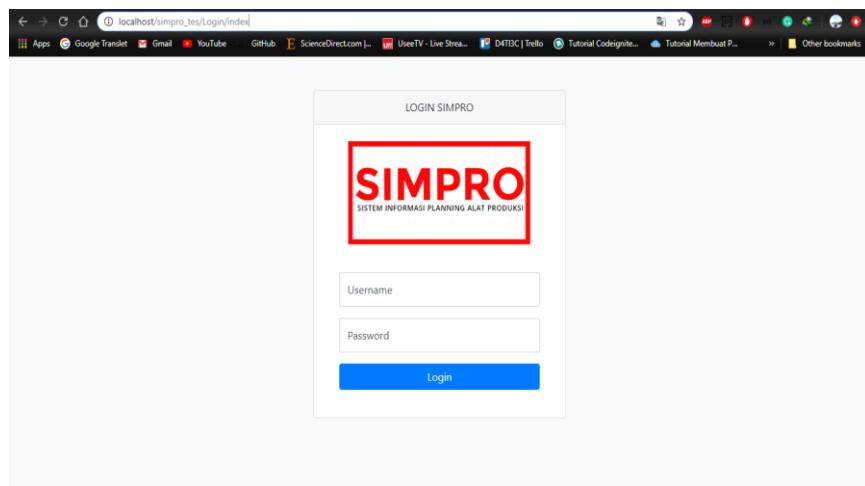
```

```
<div class="box-body">
  <form method="post" action="<?php echo base_url
() .'index.php/Sendmail_message/send' ;?>">
    <div class="form-group row">
      <label class="col-md-4 col-form-
label">Email Address</label>
      <div class="col-md-8">
        <input type="text" class="form-
control" name="email_address" placeholder="Email Ad
ress">
      </div>
    </div>
    <div class="form-group row">
      <label class="col-md-4 col-form-
label">Messages</label>
      <div class="col-md-8">
        <textarea name="message_text" placeholder="Custom T
ext Message" class="form-control"></textarea>
      </div>
    </div>
    <button type="submit" class="btn btn-primary pull-
right">Send <i class="fa fa-send"></i></button>
  </form>
</div>
</div>
</body>
</html>
```

BAB VI

HASIL ANTARMUKA SISTEM INFORMASI APPROVAL BERBASIS WEB MENGGUNAKAN FRAMEWORK CODEIGNITER DENGAN NOTIFIKASI E- MAIL

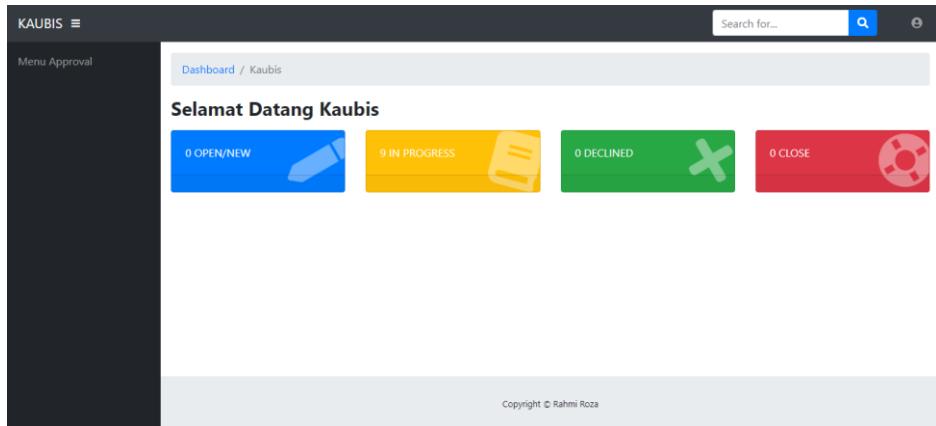
6.1 Halaman *Login*



Gambar 6.1 Halaman Login

Pada halaman *login user* akan mengisi *username* dan *password*. Jika *username* dan *password* yang diisi benar maka sistem akan otomatis mengarahkan pada halaman selanjutnya.

6.2 Halaman *Dashboard* Kaubis



Gambar 6.2 Halaman Dashboard Kaubis

Halaman ini merupakan halaman *dashboard*. Pada halaman ini ada menu *approval* untuk menyetujui permintaan yang baru masuk.

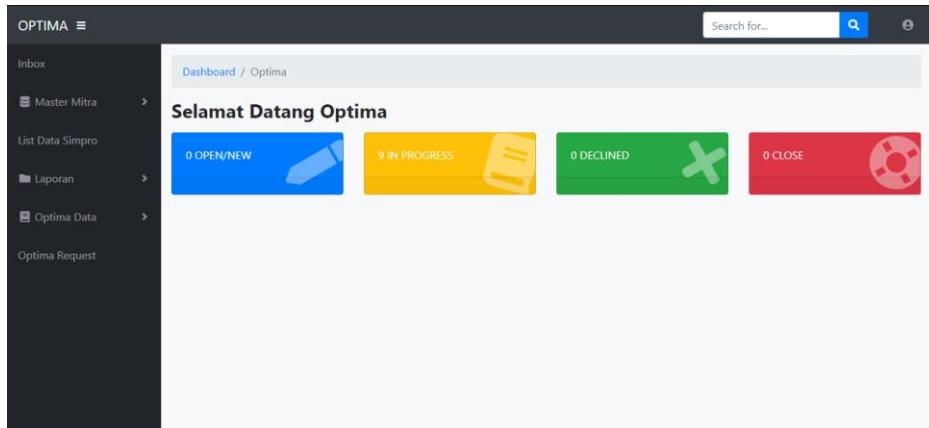
6.3 Halaman Menu *Approval* Kaubis

The screenshot shows the Kaubis approval menu with a dark sidebar on the left labeled 'Approval'. The main content area has a header 'Anda diminta untuk menyetujui permintaan di bawah ini:' and instructions 'Klik APPROVE untuk menyetujui permintaan.' Below this is a table titled 'Data Simpro' with columns: Tanggal, #ID Simpro, Wilayah, Detel, STO, UNSC, QTY ODP, Alamat, Keterangan, Status, and Aksi. The table contains three rows of data. The 'Aksi' column for each row contains a blue 'Approve' link.

Gambar 6.3 Halaman Menu Approval Kaubis

Halaman ini merupakan halaman untuk menyetujui permintaan dari data yang masuk. *User* cukup klik tulisan *approve* yang ada di ujung. Jika sudah maka otomatis permintaan akan disetujui.

6.4 Halaman *Dashboard* Optima



Gambar 6.4 Halaman Menu Dashboard Optima

Halaman ini merupakan halaman *dashboard*. Pada halaman ini ada menu *inbox*.

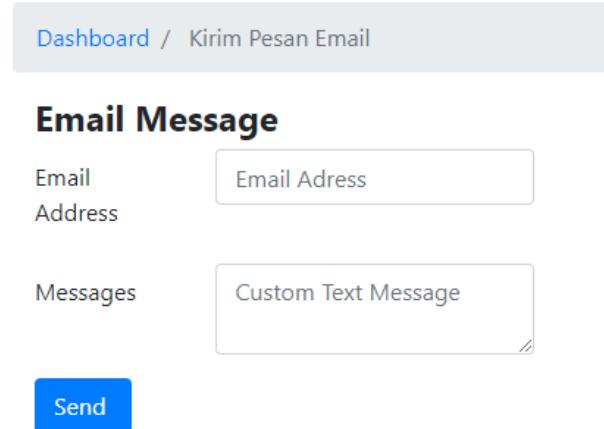
6.5 Halaman *Inbox*

A screenshot of the Optima inbox page. The sidebar on the left is identical to the dashboard. The main area shows a table titled 'Data Simpro' with columns: #ID Simpro, Time Durasi, Created, QTY ODP, Alamat, Keterangan, Status, and Kirim Pesan. There are three rows of data. Each row has a 'Kirim' button in the last column. The URL in the browser is 'localhost/simpro_tes/optima/inbox'.

Gambar 6.5 Halaman *Inbox*

Pada halaman ini *user* bisa melakukan pengiriman pesan lewat *e-mail* untuk setiap pengajuan baru. *User* cukup melakukan klik pada *button* kirim maka sistem akan otomatis mengarahkan pada form untuk pengisian pesan.

6.6 Form Halaman Kirim E-mail

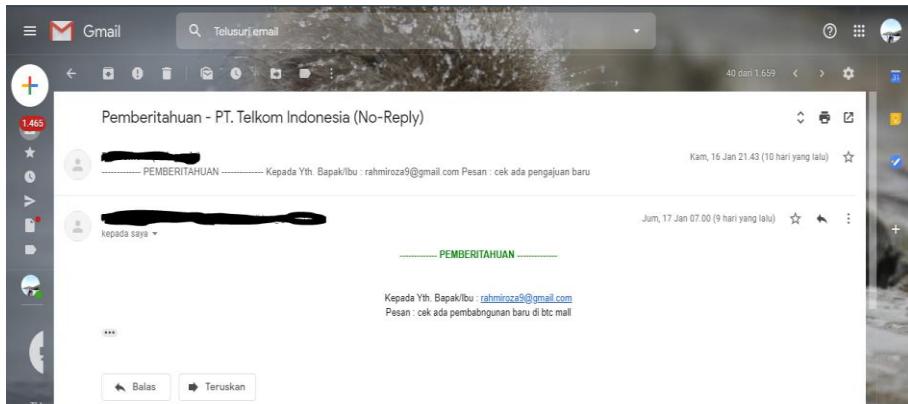


The screenshot shows a user interface for sending an email. At the top, there is a header with 'Dashboard / Kirim Pesan Email'. Below this is a section titled 'Email Message'. It contains two main input fields: 'Email Address' and 'Custom Text Message'. To the left of these fields are labels 'Email Address' and 'Messages'. Below the 'Custom Text Message' field is a text area with some placeholder text. At the bottom left is a blue 'Send' button. The overall layout is clean and modern.

Gambar 6.6 Form Kirim E-mail

Menu ini merupakan form untuk mengirimkan pesan. *User* mengisi alamat *e-mail* tujuan serta isi pesan yang diinginkan. Setelah itu klik *button send*, maka pesan akan terkirim kepada *e-mail* tujuan.

6.7 Notifikasi E-mail



Gambar 6.7 Notifikasi E-mail

Gambar di atas merupakan notifikasi *e-mail* yang masuk.

BAB VII

KESIMPULAN

Berdasarkan penelitian yang dilakukan pada PT. X dapat diambil kesimpulan yang diharapkan dapat memberikan jawaban terhadap tujuan dilakukannya penelitian sebagai berikut :

1. Dari hasil penelitian yang telah dilakukan menghasilkan sebuah Sistem Informasi *Approval* yang bertujuan untuk melakukan sebuah pengajuan pembangunan alpro (alat produksi) untuk pembangunan sebuah investasi.
2. Adanya fitur notifikasi *e-mail* yang dapat memudahkan seorang *user* mengetahui bahwa ada pengajuan baru.