

BAB II

TINJAUAN PUSTAKA

2.1 PT. Rackh Lintas Asia

PT. Rackh Lintas Asia adalah salah satu penyedia layanan *Internet Service Provider* (ISP) yang telah membangun reputasi yang kuat dalam industri telekomunikasi. Dengan kantor pusatnya yang berlokasi di Medan, perusahaan ini telah meluaskan jangkauannya dengan membuka cabang di Jakarta, memperkuat kehadirannya di pusat bisnis dan teknologi Indonesia. Sebagai penyedia layanan internet terkemuka, PT Rackh Lintas Asia tidak hanya menawarkan konektivitas internet berkualitas tinggi tetapi juga berkomitmen untuk memberikan solusi teknologi informasi yang inovatif kepada pelanggan di seluruh wilayah.

Cabang PT. Rackh Lintas Asia di Jakarta menjadi pusat strategis untuk mendukung kebutuhan pelanggan di ibu kota. Dengan infrastruktur yang canggih dan tim teknis yang terampil, cabang ini berperan penting dalam memberikan layanan yang handal dan terjangkau kepada pelanggan korporat dan individu di Jakarta. Selain itu, keberadaan PT Rackh Lintas Asia di Jakarta juga mencerminkan komitmennya untuk mendukung pertumbuhan sektor teknologi informasi dan komunikasi di wilayah ini.

Melalui rekrutmen cabang Jakarta, PT. Rackh Lintas Asia menunjukkan komitmen terhadap pengembangan sumber daya manusia lokal, meningkatkan lapangan kerja, dan mendukung pertumbuhan ekonomi di Jakarta. Dengan membangun tim yang beragam dan terampil di tingkat lokal, perusahaan dapat

meraih peluang yang lebih luas dan memberikan kontribusi positif terhadap perkembangan sektor teknologi informasi dan komunikasi di ibu kota. Oleh karena itu, rekrutmen untuk cabang Jakarta menjadi langkah strategis untuk memperkuat keberlanjutan dan dominasi pasar PT Rackh Lintas Asia dalam industri ISP di Indonesia.

2.2 Sistem Pendukung Keputusan (SPK)

Sistem Pendukung Keputusan (SPK) adalah platform berbasis komputer yang membantu menyelesaikan masalah dengan manajemen mengatasi berbagai masalah terstruktur dan tidak terstruktur dengan menggunakan data dan model yang disiapkan[1]. Menjadi alat yang penting dalam konteks pengambilan keputusan di era informasi digital, membantu organisasi dalam mengidentifikasi masalah, mengumpulkan data yang diperlukan, dan menganalisis situasi dengan cermat. Dari analisis ini, SPK dapat memberikan berbagai pilihan keputusan yang relevan yang memungkinkan para pengambil keputusan untuk membuat keputusan yang lebih terinformasi dan efektif[2].

2.2.1 Konsep Sistem Pendukung Keputusan

Konsep Sistem Pendukung Keputusan (SPK) pertama kali diungkapkan pada tahun 1971 oleh *Michael Scoot Morton* dengan istilah *Management Decision System*. Setelah itu sejumlah perusahaan, lembaga penelitian dan perguruan tinggi mulai melakukan penelitian dan membangun sistem pendukung keputusan, sehingga dari produksi yang dihasilkan dapat disimpulkan bahwa sistem ini merupakan suatu sistem yang berbasis komputer yang ditujukan untuk membantu

pengambilan keputusan dalam memanfaatkan data dan model tertentu untuk memecahkan berbagai persoalan yang tidak terstruktur[3].

Decision Support System (DSS) merupakan sistem yang memberikan fasilitas yang menyediakan informasi, permodelan, dan pemanipulasian data. Sistem itu digunakan untuk membantu pengambilan keputusan dalam situasi yang semi struktur dan situasi yang tidak terstruktur, dimana tidak ada seorang pun tahu secara pasti bagaimana keputusan seharusnya dibuat. Sistem merupakan kumpulan sub-sub sistem (elemen) yang saling berkorelasi satu dengan yang lainnya untuk mencapai tujuan tertentu. Sistem merupakan kumpulan elemen yang saling berkaitan yang bertanggung jawab memproses masukan (*input*) sehingga menghasilkan keluaran (*output*)[4].

Secara Sederhana Sistem dapat diartikan sebagai suatu kumpulan atau himpunan dari unsur, komponen, atau variabel yang terorganisir, saling berinteraksi, saling tergantung satu sama lain, terpadu[5]. Keputusan merupakan kegiatan memilih suatu strategi atau tindakan dalam pemecahan masalah tertentu. Tindakan memilih strategi atau aksi yang diyakini *supervisor* akan memberikan solusi terbaik atas sesuatu disebut pengambilan keputusan.

Suatu keputusan yang diambil untuk menyelesaikan suatu masalah dilihat dari rekonstruksi pendukung yang bisa dibagi menjadi bermacam macam klasifikasi dalam sistem pendukung keputusan guna untuk mempermudah penerapan ilmu sistem pendukung keputusan dalam berbagai aspek permasalahan. Jenis-jenis keputusan juga bisa membantu dalam menganalisis sebuah permasalahan yang akan di selesaikan dengan sistem, berikut adalah jenis-jenis keputusan:

1. Keputusan terstruktur (*structure decision*)

Keputusan terstruktur adalah keputusan yang dilakukan secara berulang-ulang dan bersikap rutin. Misalnya, keputusan pemesanan barang dan keputusan penagihan piutang.

2. Keputusan semi-terstruktur (*semistructured decision*)

Keputusan semi-terstruktur adalah keputusan yang memiliki dua sifat. Sebagian keputusan bisa atasi oleh komputer namun tetap harus dilakukan oleh pengambil keputusan. Contoh keputusan jenis ini adalah pengevaluasian kredit, penjadwalan produksi, dan pengendalian sediaan.

3. Keputusan tidak terstruktur (*unstructured decision*)

Keputusan tidak terstruktur adalah keputusan yang penanganannya rumit karena tidak terjadi berulang-ulang atau tidak selalu terjadi[1].

2.2.2 Komponen-Komponen Sistem Pendukung Keputusan

Komponen-komponen Sistem Pendukung Keputusan terdiri dari:

1. *Data Management*

Termasuk *database*, yang mengandung data yang relevan untuk berbagai situasi dan diatur oleh *software* yang disebut *Database Management System* (DBMS)[6].

2. *Model Management*

Melibatkan model finansial, statistik, *management science*, atau berbagai model kuantitatif lainnya, sehingga dapat memberikan ke sistem suatu kemampuan analitis, dan manajemen *software* yang diperlukan.

3. *Communication* (dialog subsistem)

Melalui subsistem ini, pengguna dapat berkomunikasi dan memberikan perintah pada DSS. Ini berarti menyediakan antarmuka.

4. *Knowledge Management*

Subsistem optional ini dapat mendukung subsistem lain atau bertindak sebagai komponen yang berdiri sendiri[6].

2.2.3 Tujuan Sistem Pendukung keputusan

Tujuan Sistem Pendukung Keputusan (SPK) yaitu:

1. Membantu *supervisor* membuat keputusan untuk memecahkan masalah semi terstruktur.
2. Mendukung penilaian *supervisor* bukan mencoba untuk menggantikannya.
3. Meningkatkan efektifitas pengambilan keputusan *supervisor* daripada efisiensinya.
4. Kecepatan komputasi. Komputer memungkinkan para pengambil keputusan untuk melakukan banyak komputasi secara cepat dengan biaya yang rendah.
5. Peningkatan produktivitas. Konstruksi suatu tim pengambil keputusan, terutama yang terdiri dari para ahli, dapat menimbulkan biaya yang signifikan. Solusi terkomputerisasi dapat meminimalkan ukuran tim dan memfasilitasi partisipasi anggota tim dari lokasi yang berbeda (mengurangi biaya perjalanan). Lebih lanjut, produktivitas staf pendukung, seperti analis keuangan dan hukum, dapat ditingkatkan melalui penggunaan peralatan optimisasi yang merancang strategi terbaik untuk mengelola bisnis.

6. Dukungan kualitas. Komputer Dapat meningkatkan mutu keputusan yang dihasilkan. Sebagai contoh, semakin besar akses terhadap data, semakin banyak alternatif yang dapat dievaluasi.
7. Berdaya saing. Manajemen dan pemberdayaan sumber daya perusahaan. Tekanan persaingan menyebabkan tugas pengambilan keputusan menjadi sulit.
8. Mengatasi keterbatasan kognitif dalam pemrosesan dan penyimpanan.[7]

2.2.4 Proses Pengambilan Keputusan Dalam Sistem Pendukung Keputusan

Ada tiga fase dalam proses pengambilan keputusan.

1. *Intelligence*

Tahap ini merupakan proses penelusuran dan pendeteksian ruang lingkup problematika secara proses pengenalan masalah masukan diperoleh, diproses dan diuji dalam rangka mengidentifikasi masalah.

2. *Design*

Tahap ini merupakan proses menemukan, mengembangkan dan menganalisis *alternative* tindakan yang bisa dilakukan. Tahap ini meliputi kelayakan solusi.

3. *Choice*

Pada fase ini, dilakukan seleksi antara berbagai alternatif tindakan yang potensial untuk dijalankan. Hasil dari proses seleksi tersebut selanjutnya diaplikasikan dalam tahap pengambilan keputusan.

2.2.5 Elemen Sistem Pendukung Keputusan

Elemen sistem pendukung keputusan adalah suatu pembagian ataupun entitas yang ada pada sistem pendukung keputusan itu sendiri. Secara konsep ada 3 (tiga) elemen yang terkait dengan sistem pendukung keputusan, yaitu:

1. Masalah.

Dalam sebuah sistem pendukung keputusan terdapat beberapa jenis masalah yaitu: Masalah terstruktur, masalah semi-terstruktur dan masalah tidak terstruktur.

2. Solusi.

Dalam sebuah sistem pendukung keputusan terdapat beberapa jenis solusi pemecahan masalah diantaranya yaitu: *Multi Attribute Decision Making* (MADM) seperti: metode *Simple Additive Weighting* (SAW), metode *Weight Product* (WP), metode *Analythical Hierarchy Process* (AHP), metode *Topsis* dan lain-lain. Kemudian metode *Multi Criteria Decision Making* (MCDM) seperti: metode *Promethee*, metode *Electre*, metode *Oreste*, metode *Entropi* dan lain-lain. Selain terdapat juga metode *Multi Factor Evaluation Process* (MFEP), metode *Multi Attribute Utility Theory* (MAUT), serta metode FMADM (*Fuzzy Multi Attribute Decision Making*) yang terdiri dari F-AHP, F-SAW, dan lain-lain.

3. Hasil.

Keluaran dari suatu sistem pendukung keputusan adalah keputusan yang menjadi pedoman untuk merumuskan kebijakan terkait dengan masalah yang sedang diselidiki atau dibahas. Keputusan merupakan hasil dari pemilihan strategi atau tindakan untuk menanggapi suatu masalah. Proses pemilihan strategi atau tindakan yang dianggap supervisor sebagai solusi terbaik disebut sebagai aktivitas pengambilan keputusan..[8]

2.3 Metode Yang Digunakan Dalam Pembuatan Sistem

Pada pengembangan sistem rekrutmen di PT Rackh Lintas Asia, diperlukan pendalaman algoritma sistem guna mendukung pengambilan keputusan terkait input yang dimasukkan ke dalam sistem. Metode yang diterapkan

2.3.1 Metode ARAS

Additive Ratio Assessment (ARAS) adalah sebuah metode yang digunakan untuk perangkingan kriteria, secara konsep metode ARAS ini digunakan dengan metode lain yang menggunakan konsep perangkingan seperti SAW atau TOPSIS, dimana proses penentuan ranking harus di olah kembali dengan menggunakan Metode ARAS sehingga hasil peringkat dengan metode SAW dan metode SAW+ARAS bisa berberda hasilnya.

Perhitungan Metode Additive Ratio Assessment (ARAS) dapat diartikan sebagai satu-satunya metode sistem pendukung keputusan yang dalam perangkinganya menggunakan konsep perangkingan Utility Degree, yaitu dengan membandingkan nilai indeks keseluruhan alternatif optimal terhadap nilai indeks keseluruhan setiap alternatif [9].

1. Menentukan kriteria, bobot, alternatif dan menentukan nilai alternatif dari setiap kriteria serta menentukan nilai optimal *benefit* dan *cost*.
2. Merubah nilai kriteria menjadi matriks keputusan.

$$X = \begin{bmatrix} X_{01} & X_{0j} & \dots & X_{0n} \\ X_{11} & X_{1j} & \dots & X_{1n} \\ \vdots & \vdots & \ddots & \vdots \\ X_{n1} & X_{nj} & \dots & X_{nn} \end{bmatrix} \quad (i=0, m; \dots j = 1, n) \dots\dots\dots [2.1]$$

Dimana:

m = Jumlah alternatif

n = Jumlah kriteria

X_{ij} = Nilai kriteria dari alternatif i

X_{0j} = Nilai optimal dari kriteria j

3. Menentukan nilai optimal kriteria j (X_{0j}) yaitu dengan cara:

$$X_{0j} = \frac{\text{Max}}{i} \cdot X_{ij}, \text{ If } \frac{\text{Max}}{i} \cdot X_{ij} \text{ Lebih Baik} \dots\dots\dots[2.2]$$

$$X_{0j} = \frac{\text{Min}}{i} \cdot X_{ij}, \text{ If } \frac{\text{Min}}{i} \cdot X_{ij} \text{ Lebih Baik}$$

4. Menentukan normalisasi matriks keputusan dari semua kriteria mempunyai dua cara yaitu:

Perhitungan dengan kategori Benefit Dimana X_{ij} adalah nilai normalisasi

$$X_{ij} = \frac{x_{ij}}{\sum_{i=0}^m x_{ij}} \dots\dots\dots[2.3]$$

Perhitungan dengan kategori Cost mempunyai dua cara yaitu:

$$\text{Tahap 1 : } X_{ij} = \frac{1}{x_{ij}} \dots\dots\dots[2.4]$$

$$\text{Tahap 2 : } R = \frac{x_{ij}}{\sum_{i=0}^m x_{ij}} \dots\dots\dots[2.5]$$

5. Menentukan pembobotan pada matriks yang sudah dinormalisasi:

$$D = [d_{ij}] m \times n = r_{ij} \dots\dots\dots[2.5]$$

Dimana :

W_j = bobot kriteria j

6. Menentukan Nilai dari fungsi optimalisasi (S_i)

$$S_i = \sum_{j=1}^n d_{ij} ; (i = 1, 2, \dots, m; j = 1, 2, \dots, n) \dots\dots\dots[2.6]$$

Dimana S_i merupakan fungsi optimalisasi alternatif i . Nilai terbesar adalah nilai yang terbaik dan nilai terkecil merupakan nilai yang terburuk. Nilai dan bobot kriteria yang berhubungan akan berpengaruh pada hasil akhir.

7. Langkah terakhir adalah menentukan nilai Derajat Utilitas (Peringkat) dengan menggunakan rumus:

$$K_i = \frac{S_i}{S_o} ; \dots\dots\dots[2.7]$$

Dimana hasil dari S_i dan S_o merupakan nilai kriteria optimalitas (perangkingan)

Keterangan:

K_i = nilai tingkat peringkat alternatif

S_i = nilai optimum untuk alternatif i

S_o = nilai optimum untuk alternatif optimal

2.4 UML (*Unified Modeling Language*)

Dalam perkembangan teknologi perangkat lunak, diperlukan bahasa pemodelan yang dapat digunakan untuk merancang perangkat lunak. Standarisasi bahasa tersebut juga penting agar pemodelan dapat dipahami oleh berbagai pihak di seluruh dunia. Mengingat kompleksitas kolaborasi di antara individu dengan latar belakang yang beragam, diperlukan sebuah bahasa pemodelan perangkat lunak yang dapat diakses dan dimengerti oleh banyak orang.

Unified Modelling Language (UML) adalah bahasa pemodelan untuk sistem atau perangkat lunak yang berparadigma berorientasi objek. Abstraksi konsep dasar UML terdiri dari *structural classification*, *dynamic behavior*, dan model

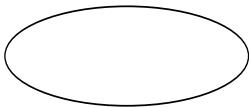



management dapat kita pahami *main concepts* sebagai *term* yang akan muncul pada saat membuat *diagram* dan *view* adalah kategori dari diagram tersebut.[10]

2.4.1 Use Case Diagram

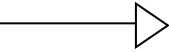
Use Case atau *diagram use case* merupakan pemodelan yang digunakan untuk menggambarkan sebuah kasus interaksi antara aktor dan sistem meliputi apa yang dapat dilakukan seorang pengguna terhadap sistem yang dijalankan.

Suatu *use case* diagram akan ditujukan untuk menyatakan visualisasi interaksi yang terjadi diantara pengguna (aktor) dengan sistem. Ada 2 elemen penting yang harus digambarkan, yaitu aktor dan *use case*. Berikut ini adalah simbol simbol *use case* diagram:

Tabel 2.1 Simbol *Use Case Diagram*

No	Nama	Simbol	Keterangan
1	<i>Use case</i>		Fungsional yang disediakan dari sistem sebagai unit-unit yang saling bertukar pesan antar unit atau <i>actor</i> .
2	Aktor / <i>actor</i>		Orang, proses atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol aktor adalah gambar orang.
3	<i>System Boundary</i>		Digambarkan dengan kotak di sekitar <i>use case</i> dan digunakan saat memberikan pilihan sistem alternatif.
4	Asosiasi/ <i>association</i>		Komunikasi antara aktor dan <i>use case</i> yang berpartisipasi pada <i>use case</i> memiliki interaksi dengan aktor.


Tabel 2.1 Simbol *Use Case Diagram* (Lanjutan)

No	Nama	Simbol	Keterangan
5	Extensi / <i>extend</i>	-- <<extend>>-->	Relasi <i>use case</i> tambahan ke semua <i>use case</i> yang ada dan berdiri sendiri
6	Generalisasi / <i>generalization</i>		Generalisasi dan spesialisasi adalah hubungan antara dua <i>use case</i> di mana satu lebih umum daripada yang lain
7	Menggunakan / <i>include / uses</i>	-- <<include>>-->	Relasi <i>use case</i> tambahan ke sebuah <i>use case</i> dimana <i>use case</i> yang ditambahkan memerlukan <i>use case</i> ini untuk menjalankan fungsinya.


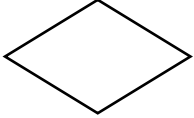
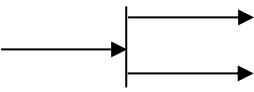
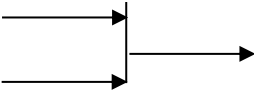

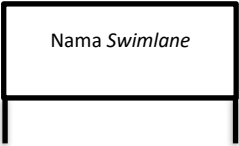
2.4.2 *Activity Diagram*

Diagram ini menggambarkan berbagai aktifitas dalam sistem yang sedang dirancang, mulai dari titik awal, melalui kondisi (*decision*) yang mungkin terjadi, kemudian sampai pada titik akhir. *Diagram* ini tidak menggambarkan perilaku/proses *internal* sebuah sistem maupun interaksi antar sub-sistem, tetapi lebih menggambarkan proses-proses dan jalur-jalur aktivitas secara umum atau global.[11]

Tabel 2.2 Simbol *Activity Diagram*

No	Nama	Gambar	Keterangan
1	Status Awal		Status awal aktifitas sistem, sebuah diagram memiliki sebuah status awal.



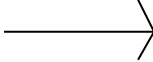

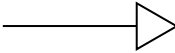
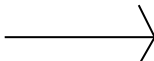
Tabel 2.2 Simbol *Activity Diagram* (Lanjutan)

No	Nama	Gambar	Keterangan
2	Aktifitas		Aktifitas yang dilakukan sistem, aktifitas biasanya diawali dengan kata kerja.
3	Percabangan/ <i>Decision</i>		Asosiasi percabangan jika ada pilihan aktifitas lebih dari satu.
4	Percabangan/ <i>fork</i>		Asosiasi percabangan lebih dari satu aktifitas dipisahkan.
5	Penggabungan/ <i>join</i>		Asosiasi penggabungan lebih dari satu aktifitas digabungkan.
6	Status akhir		Status akhir yang dilakukan sistem sebuah diagram aktifitas memiliki sebuah status akhir.
7	<i>Swimlane</i>		Memisahkan organisasi bisnis yang bertanggung jawab terhadap aktifitas yang terjadi.

2.4.3 *Class Diagram*

Diagram kelas atau *class diagram* adalah diagram yang menggambarkan struktur yang berjalan pada sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. Kelas memiliki apa yang disebut atribut dan metode atau operasi[12].

Tabel 2.3 Simbol *Class Diagram*

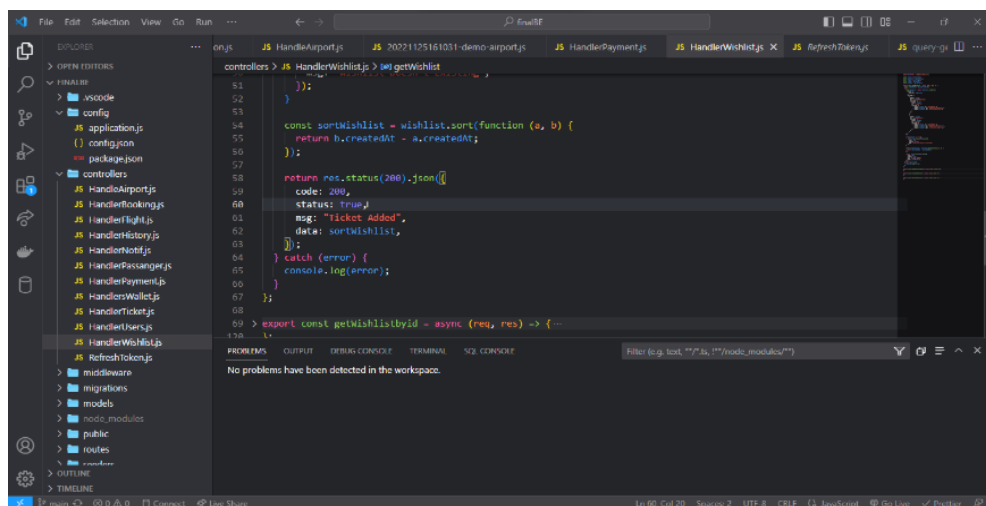
No.	Simbol	Deskripsi
1	<div style="border: 1px solid black; padding: 5px; margin: 5px auto; width: fit-content;"> Nama_kelas <hr/> + Atribut <hr/> +Operasi () </div>	Kelas pada struktur sistem
2	Antarmuka/ <i>interface</i> 	Sama dengan konsep interface dalam pemrograman berorientasi objek.
3	Asosiasi/ <i>association</i> 	Relasi antar kelas dengan makna umum, asosiasi biasanya juga disertai dengan <i>multiplicity</i>
4	Asosiasi berarah/ <i>directed association</i> 	Relasi antar kelas dengan makna kelas yang satu digunakan oleh kelas lain.
5	Agregasi/ <i>aggregation</i> 	Relasi antar kelas dengan makna semua bagian (whole-part)
6	Generalisasi 	Relasi antar kelas dengan makna generalisasi-spesialisasi (umum khusus)
7	Kebergantungan/ <i>dependency</i> 	Relasi antar kelas dengan makna kebergantungan antar kelas

2.5 Aplikasi Pendukung

Aplikasi Pendukung merujuk pada perangkat lunak atau sistem yang diciptakan untuk memberikan dukungan atau bantuan dalam berbagai kegiatan atau tugas. Aplikasi ini dirancang dengan tujuan spesifik untuk meningkatkan efisiensi, produktivitas atau kinerja dalam suatu konteks tertentu. Dalam SPK, aplikasi pendukung ini secara khusus dirancang untuk membantu proses pengambilan keputusan atau manajemen informasi. Adapun aplikasi *tool* pendukung dalam menjalankan suatu sistem antara lain:

2.5.1 Visual Studio Code

Visual Studio Code merupakan aplikasi penyuntingan kode-sumber buatan *Microsoft* untuk *Linux*, *macOS*, dan *Windows*. *VSCode* menyediakan fitur seperti penyorotan sintaksis, penyelesaian kode, kutipan kode, merefaktorkan kode, dan *Git*



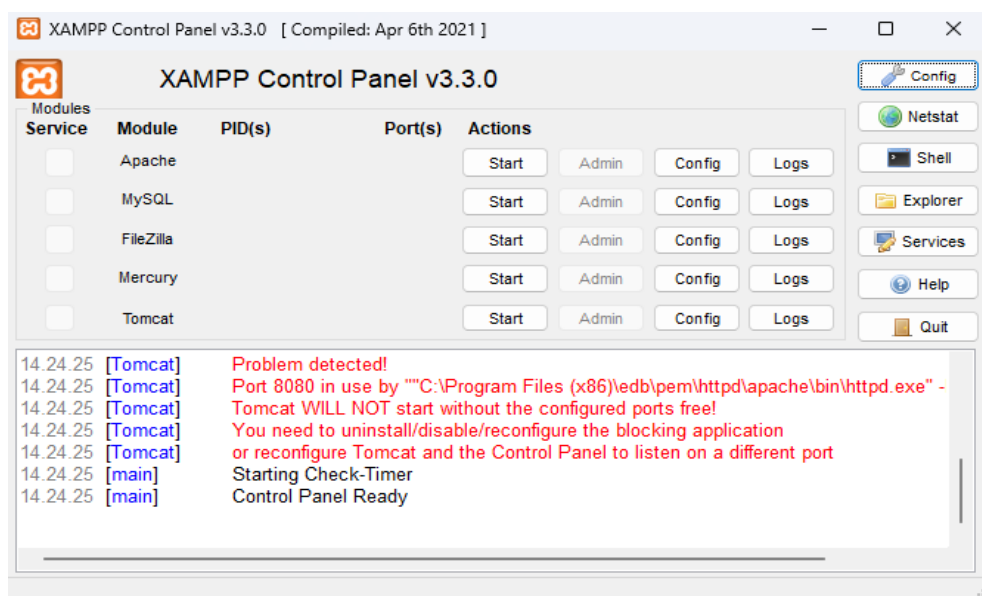
Gambar 2.1 *Visual Studio Code*

Visual Studio Code pertama kali pada tanggal 29 April 2015 oleh *Microsoft* di konferensi Build 2015. *VSCode* dibuat dengan menggunakan Electron, sebuah *framework* yang memungkinkan aplikasi web berjalan di luar browser. *VSCode*

adalah perangkat lunak gratis dan sumber terbuka yang dirilis dengan lisensi MIT. *VSCode* tersedia di *GitHub* dan dapat diunduh secara gratis. *Visual Studio Code* mendukung berbagai bahasa pemrograman, termasuk *JavaScript*, *TypeScript*, *Python*, *Java*, *C/C++*, dan lainnya. *Visual Studio Code* juga mendukung berbagai ekstensi yang dapat ditambahkan untuk menambah fitur baru atau menyesuaikan tampilan dan nuansa *Visual Studio Code*.

2.5.2 XAMPP (X-Apache, Mysql, PHP, dan Perl)

sebuah paket perangkat lunak yang terdiri dari beberapa program yang digunakan untuk membangun dan menjalankan situs web dan aplikasi web. *XAMPP* adalah singkatan dari *Apache*, *MySQL*, *PHP*, dan *Perl*. *XAMPP* adalah *software open source* berbasis *web server* yang berisi berbagai program. Aplikasi ini mendukung berbagai sistem operasi seperti *Linux*, *Windows*, *MacOS*, dan *Solaris*. Fungsi *XAMPP* adalah sebagai *server lokal/ localhost*, di dalamnya sudah mencakup program *Apache*, *MySQL* dan *PHP*[13].



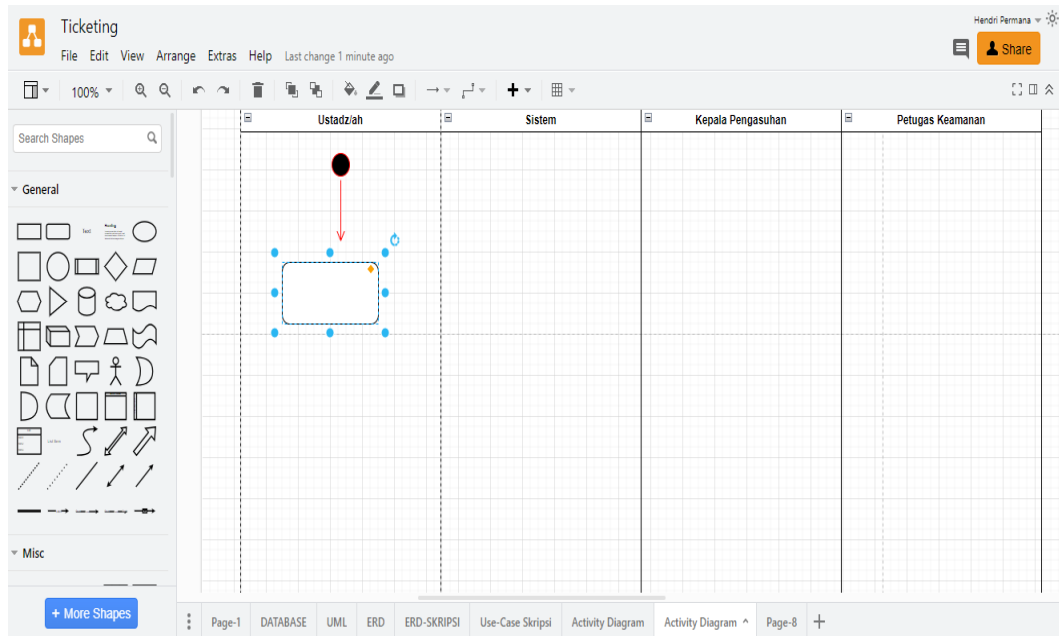
Gambar 2.2 XAMPP

XAMPP adalah pilihan yang bagus untuk pengembang web yang ingin membangun situs web dan aplikasi web lokal. *XAMPP* memungkinkan pengembang untuk menguji situs web dan aplikasi web mereka tanpa harus menghubungkannya ke internet. Dalam *XAMPP*, terdapat beberapa komponen penting yang perlu kamu ketahui. Berikut beberapa komponen utama *XAMPP* :

1. *XAMPP control panel* adalah komponen yang digunakan untuk mengelola komponen lainnya dalam *XAMPP*. Dengan menggunakan *control panel*, kamu bisa mengaktifkan fungsi *apache*, *mySQL*, *filezilla*, *config*, *netstat* dan konfigurasi *XAMPP* lainnya
2. *Htdocs* adalah Komponen *XAMPP* tersedia dalam bentuk folder, dengan folder '*htdocs*' berperan sebagai lokasi penyimpanan untuk folder dan *file* yang dapat diakses melalui peramban (*browser*). Pada penggunaan *hosting*, '*htdocs*' berfungsi sebagai folder publik. Kapasitas '*htdocs*' sejalan dengan kapasitas partisi yang digunakan. Secara umum, '*htdocs*' biasanya terletak di alamat *path* *C:\xampp\htdocs*.
3. *Config* merupakan komponen pada *XAMPP* yang berfungsi untuk mengatur pengaturan dasar. Seperti mengatur aplikasi editor teks dan browser yang akan digunakan secara default oleh aplikasi *XAMPP*.
4. *Netstat* adalah Komponen dalam *XAMPP* bertugas memeriksa ketersediaan port yang digunakan oleh *XAMPP*, memastikan tidak ada konflik dengan aplikasi lain. Jika port standar *XAMPP* sudah terpakai oleh aplikasi lain, hal ini dapat menghambat fungsi optimal aplikasi *XAMPP*. Dalam situasi ini, disarankan untuk mengganti port yang terkait dengan port lain yang masih tersedia.

2.5.3 Draw.io

Aplikasi selanjutnya, untuk membangun rancangan flow aplikasi kedalam bentuk diagram, menggunakan *Draw.io*



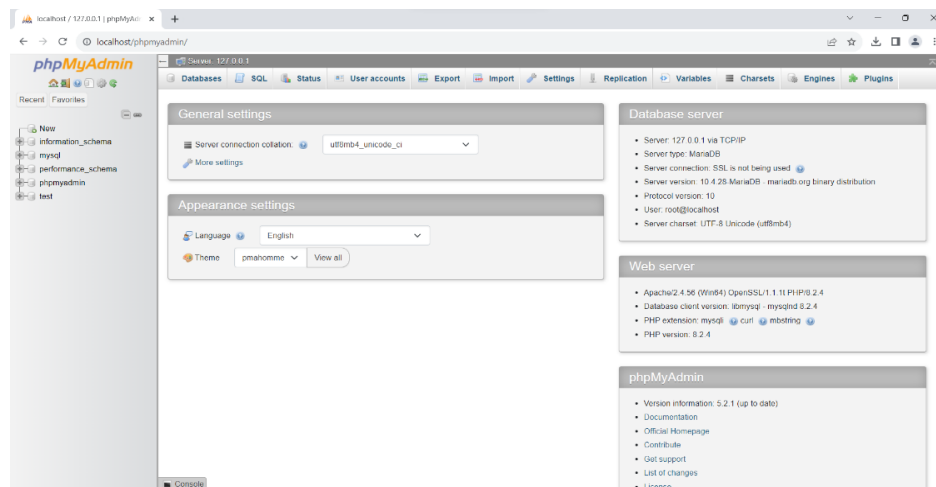
Gambar 2.3 Dashwork Draw.io

Draw.io adalah aplikasi *Draw.io* yang memungkinkan pengguna untuk membuat berbagai jenis *Draw.io*, seperti diagram alir, diagram organisasi, diagram *UML*, dan banyak lagi. Aplikasi ini dapat diakses secara gratis melalui browser web dan tidak memerlukan instalasi atau pendaftaran. Pengguna dapat menyimpan dan membaskikan diagram yang dibuat dalam berbagai format file, seperti *PNG*, *PDF*, *SVG*, atau *XML*.

2.5.4 MySQL

Adi Nugroho (2011;5), sebagaimana dikutip dalam jurnal Fery Wongso (ISSN: 1829-9822), menyatakan bahwa basis data merupakan koleksi data yang teroganisir dengan cara yang memungkinkan penyimpanan, manipulasi, dan pengambilan data oleh pengguna menjadi lebih mudah. Dalam perkembangannya,

MySQL sering disebut sebagai *SQL* yang merupakan singkatan dari *Structured Query Language*. *SQL* adalah jenis Bahasa terstruktur yang dirancang khusus untuk mengolah basis data[17]. *MySQL* adalah sistem manajemen basis data yang memiliki sifat relasional. Ini berarti bahwa data yang dikelola dalam basis data ditempatkan di beberapa tabel terpisah, yang menghasilkan proses manipulasi data yang lebih efisien. Berikut merupakan tampilan *MySQL*:



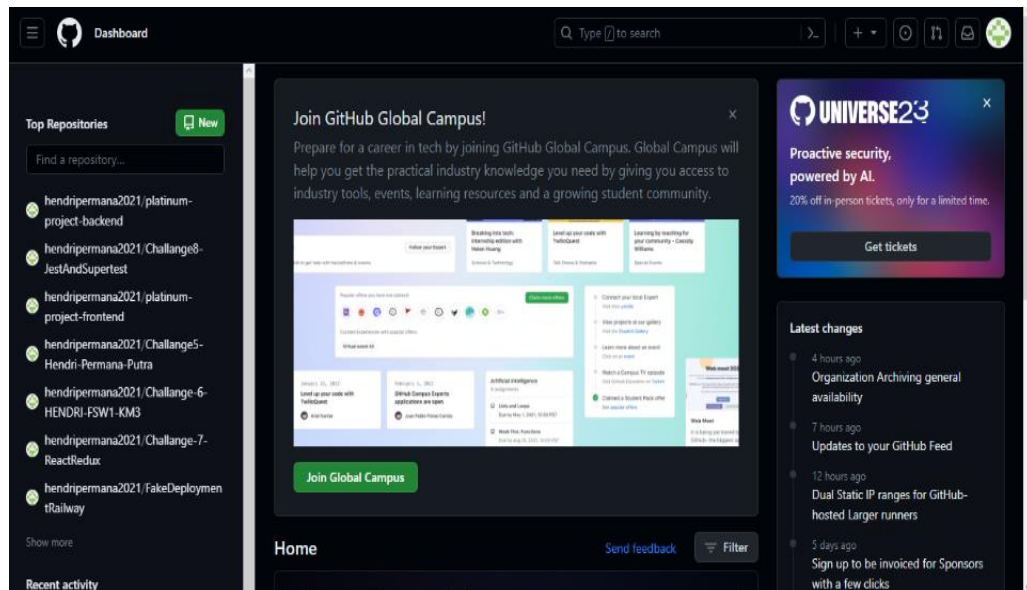
Gambar 2.4 Database *MySQL*

SQL memungkinkan pengguna untuk mengetahui lokasi atau susunan informasi dengan lebih mudah. Meskipun lebih sederhana daripada bahasa pemrograman, *SQL* memiliki tingkat kompleksitas yang lebih tinggi jika dibandingkan dengan perangkat lunak lembar kerja dan pengolah data. *SQL* adalah bahasa pemrograman yang diciptakan khusus untuk mengirimkan perintah *query* ke *database*, yaitu untuk mengakses data berdasarkan alamat tertentu.

2.5.5 Github

Dalam proses pengembangan sistem maupun pembuatan sistem, sangat diperlukan *tracking version*, pada aplikasi yang sedang dibangun, dengan tujuan untuk *monitoring updating system* apa saja yang sudah dilakukan pada sistem, serta

memudahkan *developer* untuk mengembangkan atau membangun sebuah sistem. Adapun aplikasi yang dibutuhkan adalah *GitHub*.



Gambar 2.5 Dashboard GitHub

GitHub sendiri, diambil dari 2 kata yang berbeda yaitu *Git* dan *Hub*, *Git* dikembangkan oleh Linus Torvalds pada tahun 2005, dan ini merupakan inti atau jantung GitHub. berfungsi untuk membantu developer melakukan *version control development* terhadap suatu aplikasi/software, yang berarti code base dan riwayat kode akan tersedia di komputer setiap *developer* bertujuan untuk memudahkan pembuatan *branch* dan penggabungan. Sedangkan *Hub* adalah jiwa *GitHub*. Sistem *Hub* yang ada pada *GitHub* berfungsi untuk mengubah baris perintah (*command line*), seperti *Git*, menjadi jaringan media sosial terbesar bagi para *developer*, dan memungkinkan usernya untuk berkomunikasi dengan orang-orang yang memiliki kesamaan visi dan misi.