BAB 2
LANDASAN TEORI

Pada bab ini akan menjelaskan tentang teori yang berhubungan dengan perancangan sistem sebagai teori pendukung pada saat melakukan perancangan. Selain itu, pada bab ini menjelaskan tentang spesifikasi dari komponen perangkat keras dan sistem yang digunakan pada saat perancangan sistem.

2.1 Diabetes Melitus

Diabetes melitus adalah suatu kelompok penyakit metabolik dengan karasteristik hiperglikemia yang terjadi karena sekresi insulin, gangguan kerja insulin atau keduanya, yang menimbulkan berbagai komplikasi kronik pada mata, ginjal, saraf, dan pembulu darah. Gambar luka pengidap diabetes dapat dilihat pada gambar 2.1.

Gambar 2.1 Luka Pengidap Diabetes Melitus

Faktor utama pada diabetes melitus ialah insulin, suatu hormn yang dihasilkan oleh kelompok sel ß pangkreas. Insulin memberi sinyal kepada sel tubuh agar menyerap glukosa. Insulin, bekerja dengan hormon pangkeas lain yang disebut glukagon, juga mengendalikan jumlah glukosa dalam darah. Apabila tubuh menghasilkan terlampaui sedikit insulin atau jika sel tubuh tidak

menanggapi insulin dengan tepat terjadilah diabetes. Diabetes biasanya dapat dikendalikan dengan makanan yang rendah kadar gulanya, obat yang diminum, atau suntikan insulin secara teratur [7] [8].

2.1.1 Faktor Penyebab Diabetes Melitus

Penyakit DM dapat disebabkan oleh beberapa hal, diantaranya yaitu [8] [9]:

a. Pola makan Pola makan secara berlebihan dan melebihi jumlah kalori yang dibutuhkan oleh tubuh dapat memicu timbulnya diabetes melitus (DM). Hal ini disebabkan oleh jumlah atau kadar insulin sel pangkreas mempunyai kapasitas maksimum untuk dieksresikan.
b. Obesitas Orang yang gemuk dengan berat badan melebihi 90 kg mempunyai kecenderungan lebih besar untuk terserang diabetes melitus (DM), dibadingkan dengan orang yang tidak gemuk.
c. Faktor genetik Seorang anak dapat diwarisi gen penyebab diabetes melitus (DM) dari orang tua biasanya, seseorang yang menderita diabetes melitus (DM) mempunyai anggota keluarga yag terkena juga.
d. Bahan-bahan kimia dan obat-obatan Bahan kimia tertentu dapat mengiritasi pankreas yang menyebabkan radang pangkreas. Peradangan pada pangkreas dapat menyebabkan pangkreas tidak berfungsi secra optimal dalam mensekresikan hormon yang diperlukan untuk metabolisme dalam tubuh, termaksud hormon insulin.
e. Penyakit dan infeksi pada pangkreas Mikroorganisme seperti bakteri dan virus dapat menginfeksi pangkreas sehingga menimbulkan radang pangkreas. Hal ini menyebabkan sel pada pangkreas tidak bekerja secara optimal dalam mensekresikan insulin.

2.1.2 Gejala Klinis Diabetes Melitus

Gejala klinis diabetes mellitus Gejala klinis diabetes melitus dpat digolongkan menjadi 2 golongan yaitu gejala akut dan gejala kronik [8] [10].

a. Gejala penyakit diabetes melitus dari satu penderita kependerita yang lain bervariasi, bahkan mungkin tidak menunjukan gejala apapun sampai saat

tertentu. Biasanya akan menunjukan gejala awal yaitu banyak makan (poliphagia), banyak minum (polidipsi) dan banyak kencing (poliuria). Keadaan tersebut jika tidak cepat diobati maka akan timbul gejala banyak minum, banyak berkemih, nafsu makan mulai berkurang/berat badan turun dengan cepat (turun 5-10 kg dalam waktu 3-4 minggu), mudah lelah dan bila tidak segera diobati, akan timbul rasa mual, dan penderita akan jath koma yang disebut dengan koma diabetik.

b. Gejala kronik diabetes melitus Gejala kronik yang sering dialami oleh penderita diabetes melitus adalah kesemutan, kulit terasa panas atau tertusuk-tusuk jarum, rasa tebal dikulit, kram, mudah mengantuk, mata kabur, gatal disekitar kemaluan terutama pada wanita, gigi mudah goyah dan mudah lepas, kemampuasn seksual menurun, bahkan impotensi dan para ibu hamil sering mengalami keguguran atau kematian janin dalam kandungan atau bayi lahir dengan berat badan 4 kg.

2.1.3 Klasifikasi Diabetes Melitus

Diabetes dapat diklasifikasikan menjadi 4 kategori klinis yaitu:

a. Diabetes melitus (DM) tipe 1. Diabetes melitus (DM) tipe 1 adalah penyakit autoimun kronis yang disebabkan adanya kehancuran selektif sel ß pangkreas yang memproduksi insulin. Kondisi ini ditandai dengan ditemukannya anti insulin atau antibodi sel dalam darah. Pada diabetes mellitus tipe 1 ini biasanya terjadi sebelum umur 30 tahun dan harus mendapatkan insulin dari luar [8].
b. Diabetes melitus (DM) tipe 2. Diabetes melitus (DM) tipe 2 adalah diabetes yang tidak bergantung pada insulin. Hal ini disebabkan karena diabetes mellitus tipe 2 masih mampu mensekresi insulin namun dalam kondisi yang kurang sempurna karena adanya resistensi insulin dan keadaan hiperglikemia.
c. Diabetes melitus dengan kehamilan Diabetes melitus dengan kehamilan atau diabetes melitus gestasional (DMG), merupakan penyakit diabetes

melitus yang muncul pada saat mengalami kehamilan padahal sebelumnya kadar glukosa darah selalu normal. Diabetes jenis tipe ini akan kembali normal setelah melahirkan. Faktor resiko pada diabetes mellitus dengan kehamilan ini dengan umur lebih dri 25 tahun disertai dengan riwayat keluarga dengan diabetes melitus, infeksi yang berulang , melahirkan dengan berat badan bayi lebih dari 4 kg.

d. Diabetes tipe lain disebabkan karena defek genetik fungsi sel ß pangkreas defek genetik fungsi insulin, penyakit eksorin pangkreas, endokrinopati, karena obat atau zat kimia, infeksi sindrom genetik lain yang berhubungan dengan diabetes melitus.

2.2 Gula Darah

Glukosa urine adalah gugus gula sederhana yang masih ada di urine setelah melewati proses diginjal, yang disebakan karena kekurangan hormon insulin yaitu yang mengubah glukosa menjadi glikogen. Glukosauria (Kelebihan gula didalam urine) terjadi karena nilai ambang ginjal terlampaui atau daya reabsorbsi tubulus yang menurun. Untuk pengukuran glukosa urine, reagen strip, dan enzim glukosa oksidase (GOD), Peroksidase (POD), dan zat warna [8].

Glukosa urin adalah eksresi glukosa didalam urin, dimana terjadi peningkatan pengeluaran glukosa atau gula darah melalui urin (air kemih). apabila kadar glukosa itu meningkat sementara telah diketahui bahwa ginjal hanya dapat menfiltrasi dalam jumlah tertentu maka ginjal tidak dapat menyaring semuanya dan diketahui bahwa sifat glukosa banyak menyerap air sehingga sebagian glukosa akan keluar bersama dengan urin. Dalam urin yang normal tidak ditemukan glukosa karena pada tubulus ginjal akan dilakukan proses reabsorpsi molekul glukosa untuk kembali masuk ke dalam sirkulasi darah [8].

Didalam tubuh glukosa didapat dari hasil pencernaanamilum, sukrosa, maltosa dan lactosa. Sebagai sumber energi, glukosa ditransfor dari sirkulasi darah kedalam seluruh sel-sel tubuh untuk dimetabolisme. Sebagian glukosa yang ada dalam sel diubah menjadi energi melalui proses glikolisis dan sebagian besar

lagi melalui proses glikogenesis diubah menjadi glikogen, dimana setiap saat dapat diubah kembali menjadi glukosa bila diperlukan. Jika kadar urine terlalu besar dalam darah maka dibuang melalui urine, padahal kurang dari 0,1% dari glukosa normal disaring oleh glomerulus muncul dalam urine (130 mg/24 jam). Terdapat dua penyebab glukosa urin :

a. Kadar gula darah yang terlalu tinggi. karena jika kadar gula tinggi pada darah akan berakibat pada saluran ginjal. Saluran ginjal tidak akan mampu menyerap seluruh gula tersebut sehingga akibat gula yang keluar melalui air kemih atau glukosa urin akan meningkat.
b. Kerusakan pada saluran ginjal. Kerusakan tersebut berakibat pada menurunya kemampuan ginjal untuk menyerap kembali gula. akibatnya akan ditemukan glukosa didalam urin pada saat berkemih.

Gejala Gula atau glukosa bersifat menyerap banyak air. dengan demikian, penderita glukosauria akan terjadi peningkatan volume air kemih. sehingga penderita tersebut akan mengalami sering buang air kecil, bahkan sering terbangun malam hari untuk berkemih. jika kondisi ini terus terjadi maka penderita dapat mengalami dehidrasi, lemas, sering merasa haus, dan kekurangan cairan [8].

Tabel 2.1. kadar gula darah orang normal, pre diabetes dan diabetes: *[11]*

Kadar Gula Darah	**Normal (mg/dl)**	**Prediabetes**	**Diabetes (mg/dl)**
Gula darah puasa	<100	>100 - <126	> 126
Gula darah 2 jam sesudah makan	<140	>140 - < 200	> 200

Tabel 2.2. kriteria pengendalian diabetes: *[11]*

	Kadar Baik	Kadar Sedang	Kadar Buruk
Gula darah sewaktu (mg/dl)	80-139	140-179	> 180

Gula darah puasa (mg/dl)	80-109	110-125	> 126
Gula darah 2 jam sesudah makan (mg/dl)	80-144	145-179	> 180

2.3 Urinalisis

Urinalisis adalah analisis fisik, kimia, dan mikroskopik terhadap urin. Urinalisis berguna untuk untuk mendiagnosis penyakit ginjal atau infeksi saluran kemih dan untuk mendeteksi adanya penyakit metabolik yang tidak berhubungan dengan ginjal.

Urinalisis yang akurat dipengaruhi oleh spesimen yang berkualitas. Sekresi vagina, perineum dan uretra pada wanita, dan kontaminan uretra pada pria dapat mengurangi mutu temuan laboratorium. Mukus, protein, sel, epitel, dan mikroorganisme masuk ke dalam sistem urine dari uretra dan jaringan sekitarnya. Oleh karena itu pasien perlu diberitahu agar membuang beberapa millimeter pertama urine sebelum mulai menampung urine. Pasien perlu membersihkan daerah genital sebelum berkemih. Wanita yang sedang haid harus memasukkan tampon yang bersih sebelum menampung specimen. Kadang-kadang diperlukan kateterisasi untuk memperoleh spesimen yang tidak tercemar.

Meskipun urine yang diambil secara acak (random) atau urine sewaktu cukup bagus untuk pemeriksaan, namun urine pertama pagi hari adalah yang paling bagus. Urine satu malam mencerminkan periode tanpa asupan cairan yang lama, sehingga unsur-unsur yang terbentuk mengalami pemekatan. Gunakan wadah yang bersih untuk menampung spesimen urin. Hindari sinar matahari langsung pada waktu menangani spesimen urin. Jangan gunakan urin yang mengandung antiseptik.

Lakukan pemeriksaan dalam waktu satu jam setelah buang air kecil. Penundaan pemeriksaan terhadap spesimen urine harus dihindari karena dapat mengurangi validitas hasil. Analisis harus dilakukan selambat-lambatnya 4 jam

setelah pengambilan spesimen. Dampak dari penundaan pemeriksan antara lain : unsur-unsur berbentuk dalam sedimen mulai mengalami kerusakan dalam 2 jam, urat dan fosfat yang semula larut dapat mengendap sehingga mengaburkan pemeriksaan mikroskopik elemen lain, bilirubin dan urobilinogen dapat mengalami oksidasi bila terpajan sinar matahari, bakteri berkembangbiak dan dapat mempengaruhi hasil pemeriksaan mikrobiologik dan pH, glukosa mungkin turun, dan badan keton, jika ada, akan menguap.

2.4 Pemeriksaan Glukosa Urine Metode Carik Celup

D-glukosa oleh enzim glukosa oksidase diubah menjadi D-glukonolakton dan H2O2. H2O2 yang berbentuk akan mengoksidasi kromogen membentuk senyawa berwarna coklat.

Kurang dari 0,1% dari glukosa normal disaring oleh glomerulus muncul dalam urin (kurang dari 130 mg/24 jam). Glukosuria (kelebihan gula dalam urin) terjadi karena nilai ambang ginjal terlampaui atau daya reabsorbsi tubulus yang menurun. Glukosuria umumnya berarti diabetes mellitus. Namun, glukosuria dapat terjadi tidak sejalan dengan peningkatan kadar glukosa dalam darah.

Untuk pengukuran glukosa urine, reagen strip diberi enzim glukosa oksidase (GOD), peroksidase (POD) dan zat warna. Darah disaring oleh jutaan nefron, sebuah unit fungsional dalam ginjal. Hasil penyaringan (filtrat) berisi produk-produk limbah (mis. urea), elektrolit (mis. natrium, kalium, klorida), asam amino, dan glukosa. Filtrat kemudian dialirkan ke tubulus ginjal untuk direabsorbsi dan diekskresikan; zat-zat yang diperlukan (termasuk glukosa) diserap kembali dan zat-zat yang tidak diperlukan kembali diekskresikan ke dalam urin [12].

Kurang dari 0,1% glukosa yang disaring oleh glomerulus terdapat dalam urin (kurang dari 130 mg/24 jam). Glukosuria (kelebihan gula dalam urin) terjadi karena nilai ambang ginjal terlampaui (kadar glukosa darah melebihi 160-180 mg/dl atau 8,9-10 mmol/l), atau daya reabsorbsi tubulus yang menurun.

2.4.1 Prosedur Pemeriksaan

Uji glukosa urin konvensional menggunakan pereaksi Benedict atas dasar sifat glukosa sebagai zat pereduksi. Cara ini tidak spesifik karena beberapa pereduksi lain dapat mengacaukan hasil uji [12]. gambar strip urin glukosa dapat dilihat pada gambar 2.2.

Gambar 2.2 Strip Urin Glukosa

Metode carik celup (*dipstick*) dinilai lebih bagus karena lebih spesifik untuk glukosa dan waktu pengujian yang amat singkat. Reagen strip untuk glukosa dilekati dua enzim, yaitu glukosa oksidase (GOD) dan peroksidase (POD), serta zat warna (kromogen) seperti orto-toluidin yang akan berubah warna biru jika teroksidasi. Zat warna lain yang digunakan adalah iodide yang akan berubah warna coklat jika teroksidasi. Nilai dari glukosa urin normal adalah negatif (kurang dari 50mg/dl) [12]. Prosedur uji yang akan dijelaskan disini adalah uji dipstick. Berikut adalah prosedur pemeriksaan glukosa dengan menggunakan metode caarik celup:

1. Kumpulkan spesimen acak (random)/urin sewaktu.
2. Celupkan strip reagen (dipstick) ke dalam urin.

3. Tunggu selama 60 detik, amati perubahan warna yang terjadi dan cocokkan dengan bagan warna.
4. Pembacaan dipstick dengan instrument otomatis lebih dianjurkan untuk memperkecil kesalahan dalam pembacaan secara visual.

Beberapa faktor yang mempengaruhi hasil uji dipstick adalah sebagai berikut :

1. Hasil uji positif palsu dapat disebabkan oleh : bahan pengoksidasi (hidrogen peroksida, hipoklorit, atau klorin) dalam wadah sampel urin, atau urine yang sangat asam (pH di bawah 4).
2. Hasil negatif palsu dapat disebabkan oleh : pengaruh obat (vitamin C, asam hogentisat, salisilat dalam jumlah besar, asam hidroksiindolasetat), berat jenis urine > 1,020 dan terutama bila disertai dengan pH urine yang tinggi, adanya badan keton dapat mengurangi sensitivitas pemeriksaan, infeksi bakteri.

2.5 Macam-macam Kontrol Gula Darah

1. Kadar Gula Darah Sewaktu

   Pemeriksaan kadar gula darah sewaktu adalah pemeriksaan gula darah yang dilakukan setiap waktu, tanpa ada syarat puasa dan makan. Pemeriksaan ini dilakukan sebanyak 4 kali sehari pada saat sebelum makan dan sebelum tidur sehingga dapat dilakukan secara mandiri. Pemeriksaan kadar gula darah sewaktu tidak menggambarkan pengendalian DM jangka panjang (pengendalian gula darah selama kurang lebih 3 bulan). Normalnya hasil pemeriksaan kadar gula darah sewaktu berkisar antara 80-144 mg/dl [11]. Pemeriksaan ini dilakukan untuk mengatasi permasalahan yang mungkin timbul akibat perubahan kadar gula secara mendadak.

2. Kadar Gula Darah Puasa

Pemeriksaan kadar gula darah puasa adalah pemeriksaan yang dilakukan setelah pasien berpuasa selama 8-10 jam. Pemeriksaan ini bertujuan untuk mendeteksi adanya diabetes. Standarnya pemeriksaan ini dilakukan minimal 3 bulan sekali. Kadar gula darah normal pada saat puasa adalah 70-100 mg/dl [11]. Apabila kadar gula darah pada saat puasa di atas 126 mg/dl dan 2 jam sesudah makan di atas 200 mg/dl maka seseorang diagnosis mengalami DM.

3. Kadar Gula Darah 2 Jam Setelah Makan *(Postprandial)*

   Pemeriksaan kadar postprandial adalah pemeriksaan kadar gula darah yang dilakukan saat 2 jam setelah makan. Pemeriksaan ini bertujuan untuk mendeteksi adanya diabetes.Standarnya pemeriksaan ini dilakukan minimal 3 bulan sekali. Kadar gula di dalam darah akan mencapai kadar yang paling tinggi pada saat dua jam setelah makan. Normalnya, kadar gula dalam darah tidak akan melebihi 180 mg [11]. Kadar gula darah 190 mg/dl disebut sebagai nilai ambang ginjal. Jika kadar gula melebihi nilai ambang ginjal maka kelebihan gula akan keluar bersama urin.

2.6 Internet of Things

Internet of Thing (IoT) adalah sebuah konsep sekenario dimana suatu objek yang menggunakan pemanfaatan internet yang tersambung dengan jaringan komputer secara terus menerus. adapun kemampuan seperti berbagi data kendali, dan termasuk juga pada benda dunia nyata. Bahan pangan, elektronik, peralatan apa saja, koleksi , termasuk benda hidup, yang semuanya tersambung ke jaringan lokal dan global melalui sensor tertanam dan selalu aktif [13] [14].

Interaksi antara manusia dengan manusia sudah sangat biasa sejak zaman dahulu kala. Interaksi antara manusia dengan mesin sudah biasa pula, sejak adanya penemuan teknologi seperti komputer atau gadget devices lainnya. Namun, Menurut analisa McKinsey *Global Institute, internet of things* adalah sebuah teknologi yang memungkinkan kita untuk menghubungkan mesin, peralatan, dan benda fisik lainnya dengan sensor jaringan dan aktuator untuk

memperoleh data dan mengelola kinerjanya sendiri, sehingga memungkinkan mesin untuk berkolaborasi dan bahkan bertindak berdasarkan informasi baru yang diperoleh secara independen. Sebuah publikasi mengenai *Internet of things in* 2020 menjelaskan bahwa *internet of things* adalah suatu keadaan ketika benda memiliki identitas, bisa beroperasi secara intelijen, dan bisa berkomunikasi dengan sosial, lingkungan, dan pengguna.

Teknologi *internet of things* sangat luar biasa. Jika sudah direalisasikan, teknologi ini tentu akan sangat memudahkan pekerjaan manusia. Manusia tidak akan perlu lagi mengatur mesin saat menggunakannya, tetapi mesin tersebut akan dapat mengatur dirinya sendiri dan berinteraksi dengan mesin lain yang dapat berkolaborasi dengannya. Hal ini membuat mesin-mesin tersebut dapat bekerja sendiri dan manusia dapat menikmati hasil kerja mesin-mesin tersebut tanpa harus repot-repot mengatur mereka.

Cara kerja dari *internet of things* setiap benda harus memiliki sebuah *IP Address. IP Address* adalah sebuah identitas dalam jaringan yang membuat benda tersebut bisa diperintahkan dari benda lain dalam jaringan yang sama. Selanjutnya, IP address dalam benda-benda tersebut akan dikoneksikan ke jaringan internet. Saat ini, koneksi internet sudah sangat mudah kita dapatkan. Dengan demikian, kita dapat memantau benda tersebut bahkan memberi perintah kepada benda tersebut. Setelah sebuah benda memiliki *IP address* dan terkoneksi dengan internet, pada benda tersebut juga dipasang sebuah sensor. Sensor pada benda memungkinkan benda tersebut memperoleh informasi yang dibutuhkan. Setelah memperoleh informasi, benda tersebut dapat mengolah informasi itu sendiri, bahkan berkomunikasi dengan benda-benda lain yang memiliki *IP address* dan terkoneksi dengan internet juga. Akan terjadi pertukaran informasi dalam komunikasi antara benda-benda tersebut. Setelah pengolahan informasi selesai, benda tersebut dapat bekerja dengan sendirinya, atau bahkan memerintahkan benda lain juga untuk ikut bekerja.

Dapat kita simpulkan bahwa internet of things membuat suatu koneksi antara mesin dengan mesin, sehingga mesin-mesin tersebut dapat berinteraksi dan bekerja secara independen sesuai dengan data yang diperoleh dan diolahnya

secara mandiri. Tujuannya adalah untuk membuat manusia berinteraksi dengan benda dengan lebih mudah, bahkan supaya benda juga bisa berkomunikasi dengan benda lainnya. Dengan begitu manusia dalam *internet of things* akan bertindak sebagai pengguna yang akan dilayani oleh benda-benda elektronik disekitarnya. Pengaruh ekonomi dari industri internet of things menempati posisi ketiga terbesar setelah mobile internet dan *automation o f knowledge work*. Hal ini membuktikan bahwa suatu hari nanti teknologi *internet of things* benar-benar akan berkembang dan menjadi tren di dunia [13] [15].

2.7 Jaringan Internet

Internet adalah jaringan atau sistem pada jaringan komputer yang saling berhubungan berbagai komputer untuk dapat berbagi sumber daya, komunikasi dan akses informasi [13] [16]. Dengan menggunakan Sistem *Global Transmission Control Protocol* / *Internet Protocol Suite* (TCP/IP) sebagai protokol pertukaran paket atau data (*packet switching communication protocol*) untuk melayani miliaran pengguna di seluruh dunia. Internet juga biasa dikenal sebagai *interconneted-networking* (singkatan dari *Internet*). Internet berasal dari bahasa latin, yaitu “Inter” yang memiliki arti “Antara”. Jadi, apabila digabungkan kata per kata Internet adalah jaringan antara atau penghubung.

Internet dapat diartikan sebagai jaringan komputer luas dan besar yang mendunia, yaitu menghubungkan pemakai komputer dari suatu negara ke negara lain di seluruh dunia, dimana di dalamnya terdapat berbagai sumber daya informasi dari mulai yang statis hingga dinamis dan interaktif. Dalam komunikasi ini dapat terjadi perpindahan data ataupun berbagi sumber daya secara terorganisasi di seluruh dunia melalui telepon atau satelit. Dalam skala luas.

2.8 Aplikasi Pintar (Smart Application)

Ini Pada dasarnya aplikasi adalah suatu program berbentuk perangkat lunak yang berjalan pada suatu sistem tertentu berguna sebagai hubungan antar muka user pengguna yamg berfungsi untuk membantu berbagai kegiatan [13] [13].

Dengan perkembangannya aplikasi saat berkembang dapat disebut sebagai aplikasi pintar (*Smart Apps)* dengan semakin inovatif pengembangan nya dengan

konsep pengumpulan sejumlah data dari sensor ataupun lain nya, menggunakan algoritma pembelajaran mesin dan analisis prediktif untuk membuat informasi dapat diolah dan menjadi aplikasi yang canggih selain dari pengolahan informasi memungkinkan juga mengendalikan atau mengorntrol suatu benda yang terhubung dengan aplikasi pintar tersebut. Berdasarkan penggolongannya aplikasi dapat disebut sebagai aplikasi pintar apabila memiliki unsur berikut :

1. Intelligent aplikasi pintar ini menggunakan atau mengolah data *analytics*, sebagai kemampuan agar aplikasi ini dapat belajar dan semakin pintar dengan layanan *artificial intelligent* untuk memberikan rekomendasi dan prediksi sesuai dengan fungsi dan kebutuhannya.
2. Contextual aplikasi pintar jenis ini menggunakan atau mengolah data khusus atau memanfaatkan data pribadi, sensor dan lokasi sebagai sumber daya datanya menjadikan aplikasi ini dapat mengolah dari informasi yang diberikan sensor dan memberikan keluaran sesuai dengan apa yang dibutuhkan penggunanya.
3. Proactive aplikasi pintar jenis ini memanfaatkan notifikasi *push, chat bot* dan layanan pesan untuk berinteraksi secara proaktif dengan pengguna, dengan kecerdasan buatan yang disematkan berdasarkan informasi yang ditanamkan pada aplikasi tersebut.

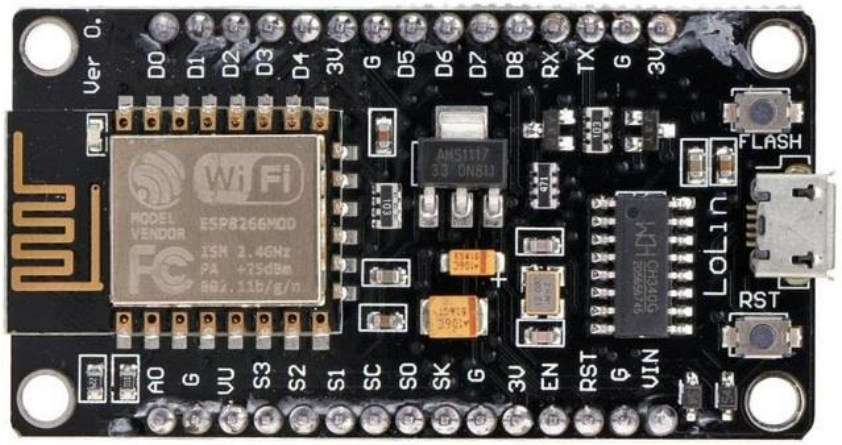
2.9 Library

Library atau pustaka merupakan suatu kumpulan *source code* yang telah ada pada compiler atau intepreter dan berfungsi untuk memudahkan pemogram untuk membuat suatu program. Biasanya untuk memanggil suatu library dapat digunakan *syntax* khusus serta dipanggil fungsi yang akan digunakan . *Library* yang disediakan dapat digunakan tergantung dari bahasa pemograman yang digunakan [13].

2.10 Modul NodeMCU

Modul WiFi NodeMCU adalah firmware interaktif berbasis LUA Espressif ESP8622 Wifi SoC. Modul ini membutuhkan daya sekitar 3.3v dengan memiliki tiga mode wifi yaitu Station, Access Point dan Both (Keduanya) [13] [17]. Modul

ini juga dilengkapi dengan prosesor, memori dan GPIO dimana jumlah pin bergantung dengan jenis **ESP8266** yang kita gunakan. Sehingga modul ini bisa berdiri sendiri tanpa menggunakan mikrokontroler apapun karena sudah memiliki perlengkapan layaknya mikrokontroler. Gambar NodeMCU dapat dilihat pada gambar 2.3.



Gambar 2.3 NodeMCU

Firmware default yang digunakan oleh perangkat ini menggunakan AT Command, selain itu ada beberapa Firmware SDK yang digunakan oleh perangkat ini berbasis opensource yang diantaranya adalah sebagai berikut:

1. **NodeMCU** dengan menggunakan basic programming LUA.
2. **MicroPython** dengan menggunakan basic programming python.
3. **AT Command** dengan menggunakan perintah perintah AT command.

Untuk pemrogramannya sendiri kita bisa menggunakan **ESPlorer** untuk Firmware berbasis **NodeMCU** dan menggunakan putty sebagai terminal control untuk AT Command. Selain itu kita bisa memprogram perangkat ini menggunakan **Arduino IDE**. Dengan menambahkan **library ESP8266** pada board manager kita dapat dengan mudah memprogram dengan basic program arduino.

Ditambah lagi dengan harga yang cukup terjangkau, kamu dapat membuat berbagai projek dengan modul ini. Maka dari itu banyak orang yang

menggunakannya modul ini untuk membuat projek Internet of Thinking (IoT). Pada perancangan alat urine analyzer untuk mendeteksi penyakit diabetes berbasis internet of things ini penulis menggunakan NodeMCU dengan library ESP8266 sebagai perangkat *Slave Receiver* yang digunakan untuk komunikasi dari mikrokontroler menuju web server.

2.11 Sensor TCS 3200

TCS3200 merupakan konverter yang diprogram untuk mengubah warna menjadi frekuensi,yang tersusun atas konfigurasi fotodiode silikondan konverter arus ke frekuensi dalam IC CMOS monolithicyang tunggal. Keluaran dari sensor ini adalah gelombang kotak (duty cycle50%) dengan frekuensi yangberbanding lurus dengan intensitas cahaya (irradiance). Gambar sensor TCS3200 dapat dilihat pada gambar 2.4.

Gambar 2.4 Sensor TCS3200

Masukan digital dan keluaran digital dari modul sensor ini memungkinkan antarmuka langsung ke mikrokontroler atau sirkuit logika lainnya. Di dalam TCS3200, konverter cahaya ke frekuensi membaca sebuah arrayfotodiode 8×8, 16 fotodiodemempunyai penyaring warna biru, 16 fotodiodemempunyai penyaring warna merah, 16 fotodiode mempunyai penyaring warna hijau, dan 16 fotodiodeuntuk warna terang tanpa penyaring. Empat tipe warna dari fotodiodediintegrasikan untuk meminimalkan efek ketidakseragaman dari insiden irradiance. Semua fotodiodedari warna yang sama terhubung secara paralel. Pin

S2 dan S3 pada modul sensor digunakan untuk memilih grup dari fotodiode (merah, hijau, biru, jernih) yang aktif [18].

2.12 Basis Data

Sistem basis data adalah sistem terkomputerisasi yang tujuan utamanya adalah memelihara data yang sudah diolah atau informasi dan membuat informasi tersedia saat dibutuhkan. Pada intinya basis data adalah media untuk menyimpan data agar dapat diakses dengan mudah dan cepat sehingga informasi tersedia saat dibutuhkan [19]. Pada buku ini menggunakan basis data relasional yang diimplementasikan dengan tabel-tabel yang saling memiliki relasi.

Sistem informasi tidak dapat dipisahkan dengan kebutuhan akan basis data apapun bentuknya, entah berupa *file* teks ataupun *Database Management System* (DBMS). Kebutuhan basis data dalam sistem informasi meliputi :

1. Memasukan, menyimpan dan mengambil data.
2. Membuat laporan berdasarkan data yang telah disimpan.

Tujuan dari buatnya table-tabel disini adalah untuk menyimpan data ke dalam tabel-tabel agar mudah diakses. Oleh karena itu, untuk merancang tabel-tabel yang akan dibuat maka dibutuhkan pola piker penyimpanan data nantinya jika dalam bentuk baris-baris data *(record)* dimana setiap baris terdiri dari beberapa kolom.

2.13 Structured Query Langage (SQL)

SQL *(Structured Query Language)* adalah bahasa yang digunakan untuk mengelola data pada RDBMS SQL awalnya dikembangkan berdasarkan teori aljabar relasional dan kalkulus.

SQL mulai berkembang pada tahun 1970an. SQL mulai digunakan sebagai standar yang resmi pada tahun 1986 oleh ANSI *(American National Standards Institute)* dan pada tahun 1987 oleh ISO *(International Organization for Standardization)* dan disebut sebagai SQL-86. Pada perkembangannya, SQL beberapa kali dilakukan revisi [13].

2.14 MySQL

MySQL adalah sebuah perangkat lunak sistem manajemen basis data SQL (*database management system*) atau DBMS yang *multithread*, MySQL adalah sebuah aplikasi *Relational Database Management* Server (RDBMS) bersifat *open source* yang memungkinkan data diakses dengan cepat oleh banyak pemakai secara bersamaan dan juga memungkinkan pembatasan akses pemakai berdasarkan privilege (hak akses) yang diberikan. Logo MySQL dapat dilihat pada gambar 2.5.

Gambar 2.5 Logo MySQL

MySQL menggunakan bahasa SQL (*structured query language)* yang merupakan bahasa standar pemograman *database.*MySQL sebenarnya merupakan turunan salah satu konsep utama dalam database sejak lama, yaitu SQL (*Structured Query Language*). SQL adalah sebuah konsep pengoperasian *database*, terutama untuk pemilihan atau seleksi dan pemasukan data, yang memungkinkan pengoperasian data dikerjakan dengan mudah secara otomastis.

MySQL dipublikasikan sejak tahun 1996, akan tetapi sebenarnya sudah dikembangkan sejak tahun 1979. MySQL telah memenangakan penghargaanLinux Journal Reader's Choice Award selama tiga tahun. MySQL sekarang tersedia di bawah lisensi open source, tapi ada juga lisensi utuk menggunakan MySQL yang bersifat komersial [13]. Berikut adalah kelebihan – kelebihan di MySQL:

1. Sederhana Skalabilitas, MSQL dapat menangani database yang besar, yang telah dibuktikan implementasinya dalam organisasi seperti Yahoo, Google, Cisco, HP, NASA dan lain sebagainya.
2. Portabilitas, MySQL dapat berjalan pada berbagai macam sistem operasi termasuk Windows, Unix, Linux, Solaris dan Mac OS. Juga dapat berjalan pada arsitektur yang berbeda, mulai dari low-end PC sampai highend mainframe.
3. Konektivitas, MySQL sepenuhnya mendukung jaringan dan dapat diakses dari mana saja di internet serta pengguna dapat mengakses21database MySQL secara bersamaan. MySQL juga menyediakan berbabagai macam API (*Aplication Program interface*) untuk mendukung konektivitas aplikasi yang ditulis dalam bahasa C, C++, Perl, PHP, Java, Phyton,C#
4. Mudah digunakan, MySQL mudah untuk digunakan dan diimplementasikan.
5. Open Source, MySQLAB membuat kode MySQL tersedia untukdigunakan setiap orang.

Beberapa perintah dasar SQL yang sering dipergunakan pada MySQL:

1. Create Database, perintah yang dipergunakan membuat database baru. Sintaks : CREATE DATABASE DATABASE_NAME.
2. Drop Database, perintah yang digunakan untuk menghapus database. Sintaks : DROP TABEL TABEL_NAME.
3. Create Tabel, perintah yang digunakan untuk membuat tabel baru. Sitntaks : Create Tabel tabel_name (create_definition).
4. Describe, perintah yang digunakan untuk mendeskripsikan tabel. Sintaks : Describe (Desc) tabel [colum].
5. Alter Tabel, perintah yang digunakan untuk menghapus tabel. Sintaks : Alter [Ignor] Tabel table_name.
6. Drop Tabel, perintah yang digunakan untuk menghapus tabel. Sintaks : Drop Tabel tabel_name [tabel_name..].

7. Sensor Delete, perintah yang digunakan untuk menghapus record dari tabel.
   Sintaks : Delete From tabel_name Where Where_definiition.
8. Select, perintah yang digunakan untuk query ke database.
   Sintaks : select * from tabel_name.
9. MySQL mendukung beberapa API yang memungkinkan aplikasi yangditulis dalam berbagai jenis bahasa pemrograman untuk berkomunikasi dengandatabase MySQL.

Berikut beberapa API yang didukung oleh MySQL:

1. C API, adalah antarmuka pemrograman utama yang memungkinkanaplikasi C/C++ untuk menghubungkan ke MySQL. Sebagian besaraplikasi klien termasuk dalam distribusi MySQL yang ditulis dalam Cdan bergantung pada API ini.
2. ODBC, MySQL mendukung Open Database Connectivity (ODBC)melalui MySQL Connector / ODBC. ODBC adalah konektivitasdatabase standar yang memungkinkan berbagai jenis aplikasi untukmenghubungkan ke berbagai jenis database.
3. JDBC, MySQL mendukung Java Database Connectivity (JDBC)melalui MySQL Connector / JDBC. JDBC adalah konektivitas database standar yang memungkinkan Java untuk menghubungkan ke berbagaijenis database.
4. PHP API, yang sekarang termasuk dengan preprocessorPHP, memungkinkan *script* PHP pada halaman web untuk berkomunikasi secara langsung dengan database MySQL. Koneksi database dan permintaan data (melalui pernyataan SQL) dikodekanlangsung dalam script PHP.

Dengan aplikasi *mySQL* bersifat *open source* memungkinkan khalayak global untuk berpartisipasi dalam pengembangan.Kelemahan MySQL dari dulu sampai saat ini adalah *feature-creep* artinya MySQL berusaha kompatibel dengan beberapa standar serta berusaha memenuhinya namun jika itu diungkapkan

kenyataannya bahwa fitur-fitur tersebut belum lengkap dan belum berperilaku sesuai standar [13].

2.15 Framework CodeIgniter

CodeIgniter merupakan sebuah framework pemrograman web dengan menggunakan bahasa php. Framework ini ditulis dengan menggunakan bahasa php versi 4 dan versi 5 oleh Rick Ellislab yang menjadi CEO Ellislab, Inc. dan dipublikasikan dengan lisensi di bawah Apache/BSD Open Source. Jadi CodeIgniter adalah framework php dan Open Source [20].

2.15.1 Manfaat

Manfaat yang dapat diambil dengan menggunakan framework CodeIgniter adalah [20]:

1. Gratis, sesuai dengan semangat Open Source untuk dapat digunakan dan dikembangkan secara bersama-sama. Dapat di-download pada alamat http://CodeIgniter.com/downloads/ secara gratis, bebas digunakan sesuai persyaratan persetujuan lisensi (lisence agreemen) yang bisa dilihat pada website tersebut [20]di atas.
2. Ditulis dengan menggunakan bahasa php 4 (untuk versi 1.x.x) dan versi 5 (untuk versi 2.x.x) sehingga mendukung pemrograman dengan bahasa php.
3. Menggunakan metode MVC sebagai prinsip kerjanya sehingga dapat digunakan untuk mengembangkan aplikasi secara efisien dan dinamis serta lebih memudahkan dalam melakukan pemeliharaan aplikasi.
4. Menggunakan URL (Uniform Resource Locator) yang sederhana, bersih, dan SEF (Search Engine Friendly).
5. Memiliki paket library yang lengkap, mendukung fungsi-fungsi database, html, web, e-mail, session, pagination dan lain-lain.

6. Dokumentasi yang lengkap dan jelas, disertakan dalam website resminya dan dapat di-download bersama-sama dengan frameworknya.
7. Komunitas, framework ini didukung oleh banyak pengguna dan pengembang, walaupun awalnya dikembangkan oleh Ellislab, Inc.
8. Bersifat portabel dan dapat dijalankan pada berbagai platform yang mendukung bahasa pemrograman php.

2.15.2 Kelemahan

Disamping kelebihan-kelebihan di atas, framework ini juga memiliki kelemahan-kelemahan yaitu [20]:

1. Longgar dalam penerapan aturan MVC, sehingga pemrograman masih diberikan kesempatan untuk melanggar kaidah-kaidah MVC.
2. Tidak mendukung konsep ORM (Object Relational Model) yaitu metode pengaksesan database dengan menggunakan relasi antar objek sehingga pemrogram tidak perlu menuliskan atau mengetahui sintaks bahasa SQL.
3. Walaupun dikembangkan oleh komunitas, namun jumlah pengembangnya tidak sebesar framework php lainnya seperti CakePHP, karena masih di bawah koordinasi Ellislab, Inc, sehingga update core engine-nya lebih lama daripada framework Open Source lainnya.
4. Sebagai framework Open Source, CodeIgniter tidak menyediakan dukungan (support) secara khusus kecuali melalui forum pengguna.

2.15.3 Prinsip Kerja

Prinsip kerja utama framework CodeIgniter terletak pada file index.php yang diletakkan pada direktori root aplikasi. File ini akan memicu dan mengarahkan permintaan layanan halaman web ke dalam tubuh framework CodeIgniter. Mekanismenya adalah sebagai berikut [20]:

1. File index.php bertindak sebagai pengendali utama yang berfungsi memuat kode script utama yang berfungsi menjalankan CodeIgniter.

2. Selanjutnya, modul routing berfungsi menerima permintaan layanan HTTP untuk menentukan arah eksekusi script yang akan dilaksanakan.
3. Jika konfigurasi cache tersedia, maka sistem langsung mengeksekusi untuk ditampilkan di halaman web.
4. Permintaan-permintaan layanan HTTP dan data-data dari form yang dikirimkan ke server, akan di-filter dan diamankan oleh modul security.
5. Selanjutnya data dikirmkan ke modul Controller yang akan mengendalikan akses ke modul Model, Library, Helper, Plugin, dan modul-modul sumber daya lainnya.
6. Kemudian Controller akan mengirimkan data ke modul View untuk ditampilkan ke halaman web. Jika konfigurasi caching diaktifkan, maka sebelum tampilan ini dikirimkan ke web untuk ditampilkan ke browser, maka tampilan ini akan di-cache sehingga permintaan yang sama dapat dilakukan dengan lebih cepat.

2.16 Web Service

Web service adalah standar yang digunakan untuk melakukan pertukaran data antar aplikasi atau sistem, karena aplikasi yang melakukan pertukaran data bisa ditulis dengan bahasa pemrograman yang berbeda atau berjalan pada *platform* yang berbeda. *Web service* digunakan sebagai suatu fasilitas yang disediakan oleh suatu *website* untuk menyediakan layanan (dalam bentuk informasi) kepada sistem lain, sehingga sistem lain dapat berinteraksi dengan sistem tersebut melalui layanan-layanan (*service*) yang disediakan oleh suatu sistem yang menyediakan *web service. Web service* menyimpan data informasi dalam format XML, sehingga data ini dapat diakses oleh sistem lain walaupun berbeda platform, sistem operasi, maupun bahasa compiler [13].

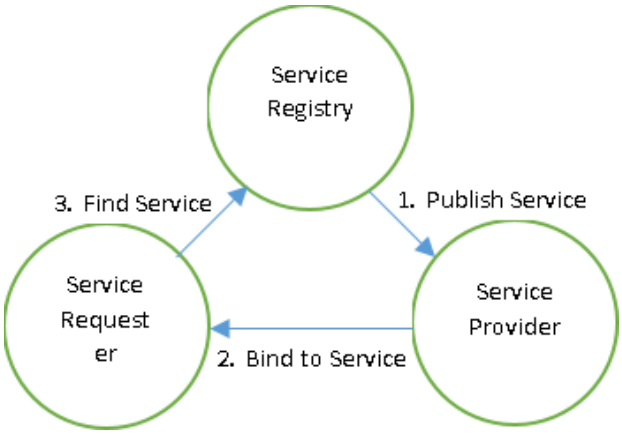
Web service bertujuan untuk meningkatkan kolaborasi antar pemrogram dan perusahaan, yang memungkinkan sebuah fungsi di dalam *Web Service* dapat dipinjam oleh aplikasi lain tanpa perlu mengetahui detil pemrograman yang

terdapat di dalamnya. Beberapa alasan mengapa digunakannya *web service* adalah sebagai berikut :

1. Sensor merupakan jenis tranduser yang digunakan sebagai pengubah besaran mekanis, magnetis, panas, sinar, dan kimia menjadi suatu tegangan dan arus listrik. Sensor biasanya digunakan untuk melakukan pengukuran atau untuk melakukan pengendalian. Berikut ini adalah jenis-jenis sensor yang digunakan.
2. Web service memiliki kemudahan dalam proses deployment-nya, karena tidak memerlukan registrasi khusus ke dalam suatu sistem operasi. *Web service* cukup di-*upload* ke *web server* dan siap diakses oleh pihak-pihak yang telah diberikan otorisasi.
3. Web service berjalan di port 80 yang merupakan protokol standar HTTP, dengan demikian *web service* tidak memerlukan konfigurasi khusus di sisi *firewall.*

2.16.1 Arsitektur Web Service

Web service memiliki tiga entitas dalam arsitekturnya, yaitu: *Service Requester* (peminta layanan), *Service Provider* (penyedia layanan), *Service Registry (*daftar layanan) yang setiap entitas arsitektur tersebut ikut berperan penting dalam pembangunan arsitektur *Web Service* [13]. Berikut ini adalah gambar arsitetktur dari *web service* dapat dilihat pada gambar 2.6.



Gambar 2.6 Arsitektur Web Service

a. *Service Provider*: Berfungsi untuk menyediakan layanan/*service* dan mengolah sebuah *registry* agar layanan-layanan tersebut dapat tersedia.
b. *Service Registry*: Berfungsi sebagai lokasi central yang mendeskripsikan semua layanan/*service* yang telah di-register.
c. Sensor *Service Requestor*: Peminta layanan yang mencari dan menemukan layanan yang dibutuhkan serta menggunakan layanan tersebut.

2.16.2 Operasi-Operasi Web Service

Secara umum, *web service* memiliki tiga operasi yang terjadi di dalamnya, yakni meliputi :

1. *Publish/Unpublish*: Menerbitkan atau menghapus layanan ke dalam atau dari *registry.*
2. *Find*: *Service requestor* mencari dan menemukan layanan yang dibutuhkan.
3. *Bind: Service requestor* setelah menemukan layanan yang dicarinya, kemudian melakukan binding ke *service provider* untuk melakukan interaksi dan mengakses layanan/*service* yang disediakan oleh *service provider.*

2.17 Browser Web

Browser web adalah *software* yang digunakan untuk menampilkan informasi dari server *web*. *Software* ini kini telah dikembangkan dengan menggunakan user interface grafis, sehingga pemakai dapat dengan melakukan *point* dan *click* untuk pindah antar dokumen. Lynx adalah *web browser* yang masih menggunakan mode teks, yang akibatnya adalah tidak ada gambar yang dapat ditampilkan.

Lynx ini ada pada lingkungan DOS dan Unix. Akan tetapi perkembangan baru web browser dengan GUI. Dapat dikatakan saat ini ada banyak web browser GUI, diantaranya Internet Explorer, Google Chrome, Mozilla Firefox dan Opera Beberapa browser tersebut kini bersaing untuk merebut pemakainya, dengan

berusaha danmendekati standar dokumen HTML yang direkomendasikan oleh W3 [13].

2.18 Web

Web atau lebih dikenal dengan sebutan *website* merupakan halaman situs sistem informasi yang dapat diakses secara cepat. ini didasari dari adanya perkembangan teknologi informasi tentang internet dan komunikasi. Melalui perkembangan teknologi informasi, tercipta suatu jaringan antar komputer yang salingberkaitan. Jaringan yang dikenal dengan istilah internet secara terus-menerus menjadi pesan–pesan elektronik [13].

Pada umumnya, halaman situs *web* berupa dokumen yang ditulis dengan format HTML (*Hyper Text Markup Language*) dan dapat diakses melalui HTTP (*Hyper Text Transfer Protocol*). HTTP adalah protokol pengirim informasi dari *server* sebuah *website* yang akan ditampilkan kepada end user melalui *web browser*.

Alamat sebuah *website* dapat menggunakan sebuah *domain* atau *subdomain*. Situs *web* harus ditempatkan pada sebuah hosting yang tergabung ke dalam WWW (*World Wide Web*) agar dapat diakses oleh orang-orang.

2.19 PHP

PHP merupakan singkatan dari PHP *Hypertext Preprocessor* yang digunakan sebagai bahasa *script server-side* atau skrip yang dijalankan disisi server dalam pengembangan web yang disisipkan pada dokumen HTML. PHP merupakan *software open source* yang disebarkan dan dilisensikan secara gratis. Selain itu Keuntungan menggunakan PHP, kode yang menyusun program tidak perlu dibagikan kepemakai, yang berarti bahwa kerahasiaan kode dapat dilindungi. Secara khusus, PHP dirancang untuk membentuk *web* dinamis. Artinya, ia dapat membentuk suatu tampilan berdasarkan permintaan terkini. Misalnya, anda bisa menampilkan isi database ke halaman *web*. Pada prinsipnya,

PHP mempunyai fungsi yang sama dengan skrip-skrip seperti ASP (Active ServerPage), Cold Fusion ataupun Perl [13].

Kelebihan dari PHP adalah sebagai berikut :

a. Bahasa pemograman PHP adalah sebuah bahasa *script* yang tidak melakukan sebuah kompilasi dalam penggunanya.
b. Web *server* yang mendukung PHP dapat ditemukan dimana-mana dari mulai apache, IIS, Lightpd, nginx, hingga Xitami dengan konfigurasi lebih mudah.
c. Dalam sisi pengembangan lebih mudah, karena banyaknya milis-milis dan *developer* yang siap membantu pengembangan.
d. Dalam sisi pemahaman, PHP adalah bahasa *scripting* yang paling mudah karena memiliki referensi yang banyak.
e. PHP adalah bahasa *open source* yang dapat digunakan dibeberapa mesin (Linux, Unix, Macintosh, Windows) dan dapat dijalankan secara *runtime* melalui *console* serta juga dapat menjalankan perintah-perintah sistem.

2.20 Cascading Style Sheet (CSS)

Cascading Style Sheet (CSS) merupakan aturan untuk mengatur beberapa komponen dalam sebuah *web* sehingga akan lebih terstruktur dan seragam. CSS bukan merupakan bahasa pemrograman.

Sama halnya *styles* dalam aplikasi pengolahan kata seperti Microsoft Word yang dapat mengatur beberapa *style*, misalnya *heading*, subbab, *bodytext*, *footer*, *images*, dan *style* lainnya untuk dapat digunakan bersama-sama dalam beberapa berkas (*file*). Pada umumnya CSS dipakai untuk memformat tampilan halaman *web* yang dibuat dengan bahasa HTML dan XHTML. CSS dapat mengendalikan ukuran gambar, warna bagian tubuh pada teks, warna tabel, ukuran *border*, warna *border*, warna *hyperlink*, warna *mouse over*, spasi antar paragraf, spasi antar teks, *margin* kiri, kanan, atas, bawah, dan parameter lainnya. CSS adalah bahasa *style sheet* yang digunakan untuk mengatur tampilan dokumen. Dengan adanya CSS

memungkinkan untuk menampilkan halaman yang sama dengan format yang berbeda.

Untuk saat ini terdapat tiga versi CSS, yaitu CSS1, CSS2, dan CSS3. CSS1 dikembangkan berpusat pada pemformatan dokumen HTML, CSS 2 dikembangkan untuk memenuhi kebutuhan terhadap format dokumen agar bisa ditampilkan di *printer*, sedangkan CSS3 adalah versi terbaru dari CSS yang mampu melakukan banyak hal dalam desain *website*. CSS2 mendukung penentuan posisi konten, *downloadable*, huruf *font*, tampilan pada tabel / *table layout* dan media tipe untuk *printer*. Kehadiran versi CSS yang kedua diharapkan lebih baik dari versi pertama dan kedua.

CSS3 juga dapat melakukan animasi pada halaman *website*, diantaranya animasi warna hingga animasi 3D. Dengan CSS 3 desainer lebih dimudahkan dalam hal kompatibilitas websitenya pada *smartphone* dengan dukungan fitur baru yakni *media query*. Selain itu, banyak fitur baru pada CSS3 seperti: *multiple background*, *border-radius*, *drop-shadow*, *border-image*, CSS *Math*, dan CSS *Object Model* [13].

2.21 Unified Modelling Language

Pada perkembangan teknologi perangkat lunak, diperlukan adanya bahasa yang digunakan untuk memodelkan perangkat lunak yang akan dibuat dan perlu adanya standarisasi agar orang di berbagai negara dapat mengerti pemodelan perangkat lunak. untuk menceritakan sebuah ide dengan tujuan untuk memahami hal yang tidaklah mudah, oleh karena itu diperlukan sebuah bahasa pemodelan perangkat lunak yang dapat dimengerti oleh banyak orang.

Unified Modelling Language (UML) adalah sebuah "bahasa" yg telah menjadi standar dalam industri untuk visualisasi, merancang dan mendokumentasikan sistem piranti lunak. UML menawarkan sebuah standar untuk merancang model sebuah sistem. Dengan menggunakan UML kita dapat membuat model untuk semua jenis aplikasi piranti lunak, dimana aplikasi tersebut dapat berjalan pada piranti keras, sistem operasi dan jaringan apapun, serta ditulis

dalam bahasa pemrograman apapun. Tetapi karena UML juga menggunakan class dan operation dalam konsep dasarnya, maka ia lebih cocok untuk penulisan piranti lunak dalam bahasabahasa berorientasi objek seperti C++, Java, C# atau VB.NET. Walaupun demikian, UML tetap dapat digunakan untuk modeling aplikasi prosedural dalam VB atau C. Seperti bahasa-bahasa lainnya, UML mendefinisikan notasi dan syntax/semantik.

Notasi UML merupakan sekumpulan bentuk khusus untuk menggambarkan berbagai diagram piranti lunak. Setiap bentuk memiliki makna tertentu, dan UML syntax mendefinisikan bagaimana bentuk-bentuk tersebut dapat dikombinasikan. Notasi UML terutama diturunkan dari 3 notasi yang telah ada sebelumnya: Grady Booch OOD (Object-Oriented Design), Jim Rumbaugh OMT (Object Modeling Technique), dan Ivar Jacobson OOSE *(Object-Oriented Software Engineering).*

Sejarah UML sendiri cukup panjang. Sampai era tahun 1990 seperti kita ketahui puluhan metodologi pemodelan berorientasi objek telah bermunculan di dunia. Diantaranya adalah: metodologi booch, metodologi coad, metodologi OOSE, metodologi OMT, metodologi shlaer-mellor, metodologi wirfs-brock, dsb. Masa itu terkenal dengan masa perang metodologi (method war) dalam pendesainan berorientasi objek. Masing-masing metodologi membawa notasi sendiri-sendiri, yang mengakibatkan timbul masalah baru apabila kita bekerjasama dengan group/perusahaan lain yang menggunakan metodologi yang berlainan. Maka dibentuklah sebuah standar dari permodelan perangkat lunak yaitu UML [13] [21].

2.21.1 Konsep Dasar Berorientasi Objek

Pendekatan berorientasi objek merupakan suatu teknik atau cara pendekatan dalam melihat permasalahan dan sistem (sistem perangkat lunak, sistem informasi, atau sistem lainnya). Pendekatan berorientasi objek akan memandang sistem yang akan dikembangkan sebagai suatu kumpulan objek yang berkorespondensi dengan objek-objek dunia nyata.

Ada banyak cara untuk mengabstraksikan dan memodelkan objek-objek tersebut, mulai dan abstraksi objek, kelas, hubungan antar kelas sampai abstraksi sistem. Saat mengabstraksikan dan memodelkan objek, data dan proses-proses yang dipunyai oleh objek akan dienkapsulasi (dihubungkus) menjadi suatu kesatuan.

Dalam rekayasa perangkat lunak, konsep pendekatan berorientasi objek dapat diterapkan pada tahap analisis, perancanggan, pemrograman, dan pengujian perangkat lunak. Ada berbagai teknik yang dapat digunakan pada masing-masing tahap tersebut, dengan aturan dan alat bantu pemodelan tertentu.

Sistem berorientasi objek merupakan sebuah sistem yang dibangun dengan berdasarkan metode berorientasi objek adalah sebuah sistem yang komponennya dibungkus (dienkapsulasi) menjadi kelompok daata dan fungsi. Setiap komponen dalam sistem tersebut dapat mewarisi atribut dan sifat dan komponen lainnya, dan dapat berinteraksi satu sama lain [13] [22].

Berikut ini adalah beberapa konsep dasar yang harus dipahami tentang metodologi berorientasi objek :

1. Kelas *(class)*

   Kelas adalah sekumpulan objek-objek dengan karakteristik yang sama. Kelas merupakan definisi statis dan himpunan objek yang sama yang mungkin lahir atau diciptakan dan kelas tersebut. Sebuah kelas akan mempunyai sifat (atribut), kelakuan (metode/operasi), hubungan (*relationship*) dan arti. Suatu kelas dapat diturunkan dan kelas yang lain, dimana atribut dan kelas semula dapat diwariskan ke kelas yang baru. Secara teknik kelas adalah sebuah struktur dalam pembuatan perangkat lunak. Kelas merupakan bentuk struktur pada kode program yang menggunakan metodologi berorientasi objek.

2. Objek *(object)*

   Objek adalah abstraksi dari sesuatu yang mewakili dunia nyata benda, manusia, satua organisasi, tempat, kejadian, struktur, status, atau hal-hal lain yang bersifat abstra. Objek merupakan suatu entitas yang mampu menyimpan informasi (status) dan mempunyai operasi (kelakuan)

yang dapat diterapkan atau dapat berpengaruh pada status objeknya. Objek mempunyai siklus hidup yaitu diciptakan, dimanipulasi, dan dihancurkan. Secara teknis, sebuah kelas saat program dieksekusi makan akan dibuat sebuah objek. Objek dilihat darisegi teknis adalah elemen pada saat *runtime* yang akan diciptakan, dimanipulasi, dan dihancurkan saat eksekusi sehinga sebuah objek hanya ada saat sebuah program dieksekusi. Jika masih dalam bentuk kode, disebut sebagai kelas jadi pada saat *runtime* (saat sebuah program dieksekusi), yang kita punya adalah objek, di dalam teks program yang kita lihat hanyalah kelas.

3. Metode (*method*)

   Operasi atau metode pada sebuah kelas hampir sama dengan fungsi atau prosedur pada metodologi struktural. Sebuah kelas boleh memiliki lebih dari satu metode atau operasi. Metode atau operasi yang berfungsi untuk memanipulasi objek itu sendiri. Operasi atau metode merupakan fungsi atau transformasi yang dapat dilakukan terhadap objek atau dilakukan oleh objek. Metode atau operasi dapat berasal dari *event*, aktifitas atau aksi keadaan, fungsi, atau kelakuan dunnia nyata. Contoh metode atau operasi misalnya *Read, Write, Move, Copy,* dan sebagainya.

4. Atribut (*atribute*)

   Atribut dari sebuah kelas adalah variabel global yang dimiliki sebuah kelas. Atribut dapat beruoa nilai atau elemen-elemen data yang dimiliki oleh objek dalam kelas objek. Atribut dipunyai secara individual oleh sebuah objek, misalnya berat, jenis, nama, dan sebagainya.

5. Abstraksi (*abstraction*)

   Prinsip untuk merepresentasikan dunia nyata yang kompleks menjadi satu bentuk model yang sederhana dengan mengabaikan aspek-aspek lain yang tidak sesuai dengan permasalahan.

6. Enkapsulasi (*encapsulation*)

   Pembungkusa atribut data dan layanan (operasi-operasi) yang dipunyai objek untuk menyembunyikan implementasi dan objek sehingga objek lain tidak mengetahui cara kerjanya.

7. Pewarisan (*inheritance*)

   Mekanisme yang memungkinkan satu objek mewarisi sebagian atau seluruh definisi dan objek lain sebagai bagian dan dirinya.

8. Antarmuka (*interface*)

   Antarmuka sangat mirip dengan kelas, tapi tanpa atribut kelas dan memiliki metode yang dideklarasikan tanpa isi. Deklarasi metode pada sebuah *interface* dapat diimplementasikan oleh kelas lain.

9. *Reusability*

   Pemanfaatan kembali objek yang sudah didefinisikan untuk suatu permasalahan pada permasalahan lainnya yang melibatkan objek tersebut.

10. Generalisasi dan Spesialisasi

    Menunjukkan hubungan antara kelas dan objek yang umum dengan kelas dan objek yang khusus. Misalnya kelas yang lebih umum (generalisasi) adalah kendaraan darat dan kelas khususnya (spesialisasi) adalah mobil, motor, dan kereta.

11. Komunikasi Antar Objek

    Komunikasi antarobjek dilakukan lewat pesan (*message*) yang dikirim dari satu objek ke objek lainnya.

12. Polimorfisme (*polymorphism*)

    Kemampuan suatu objek digunakan di banyak tujuan yang berbeda dengan nama yang sehingga menghemat baris program.

13. *Package*

    Package adalah sebuah kontainer atau kemasan yang dapat digunakan untuk mengelompokkan kelas-kelas sehingga memungkinkan beberapa kelas yang bernama sama disimpan dalam *package* yang berbeda.

2.21.2 Pengenalan Unified ModellingLanguage (UML)

Pada perkembangan teknologi perangkat lunak, diperlukan adanya bahasa yang digunakan untuk memodelkan perangkat lunak yang akan dibuat dan perlu adanya standarisasi agar orang di berbagai negara dapat mengerti pemodelan

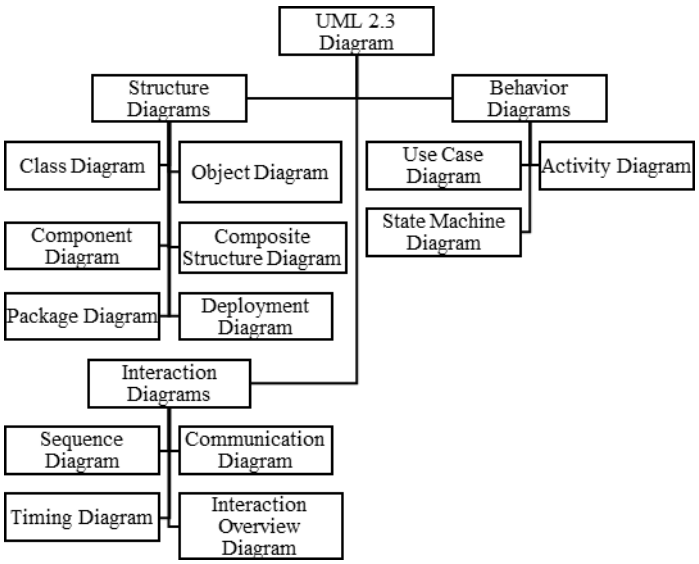
perangkat lunak. Seperti yang kita ketahui bahwa menyatukan banyak kepala untuk menceritakan sebuah ide dengan tujuan untuk memahami hal yang tidaklah mudah, oleh karena itu diperlukan sebuah bahasa pemodelan perangkat lunak yang dapat dimengerti oleh banyak orang.

Pada perkembangan teknil pemrograman berorientasi objek, munculah sebuah standarisasi bahasa pemodelan untuk pembangunan perangkat lunak yang dibangun dengan menggunakan teknik pemrograman berorientasi objek, yaitu *Unified Modeling Language* (UML). UML muncul karena adanya kebutuhanpemodelan voisual untu menspesifikasikan, menggambarkan, membangun, dan dokumentasi dari sistem perangkat lunak. UML merupakan bahasa *visual* untuk pemodelan dan komunikasi mengenai sebuah sistem dengan menggunakan digaram dan teks-teks pendukung.UML hanya berfungsi untuk melakukan pemodelan. Jadi penggunaan UML tidak terbatas pada metodologi tertentu, meskipun pada kenyataannya UML paling banyak digunakan pada metodologi berorientasi objek.

Seperti yang kita ketahui di dunia sistem informasi yang tidak dapat dibakukan, semua tergantung kebutuhan, lingkunan dan konteksnya. Begitu juga dengan perkembangan penggunaan UML bergantung pada level abstraksi penggunaannya. Jadi, belum tentu pandangan yang perbeda dalam penggunaan UML adalah suatu yang salah, tapi perlu ditelaah dimanakah UML digunakan dan hal apa yang ingin digambarkan. Secara analogi jika dengan bahasa yang digunakan sehari-hari, belum tentu penyampaian bahasa dengan puisi adalahhal yang salah. Sistem informasi bukanlah ilmu pasti, maka jika ada banyak perbedaan dan interpretasi di dalam bidang sistem informasi [13] [22].

2.21.3 Diagram UML

Pada UML 2.3 terdiri dari 13 macam diagram yang dikelompokkan dalam 3 kategori. Pembagian kategori dan macam-macam diagram tersebut dapat dilihat pada gambar 2.7.



Gambar 2.7 Diagram UML

Berikut ini penjelasan singkat dari pembagian kategori tersebut :

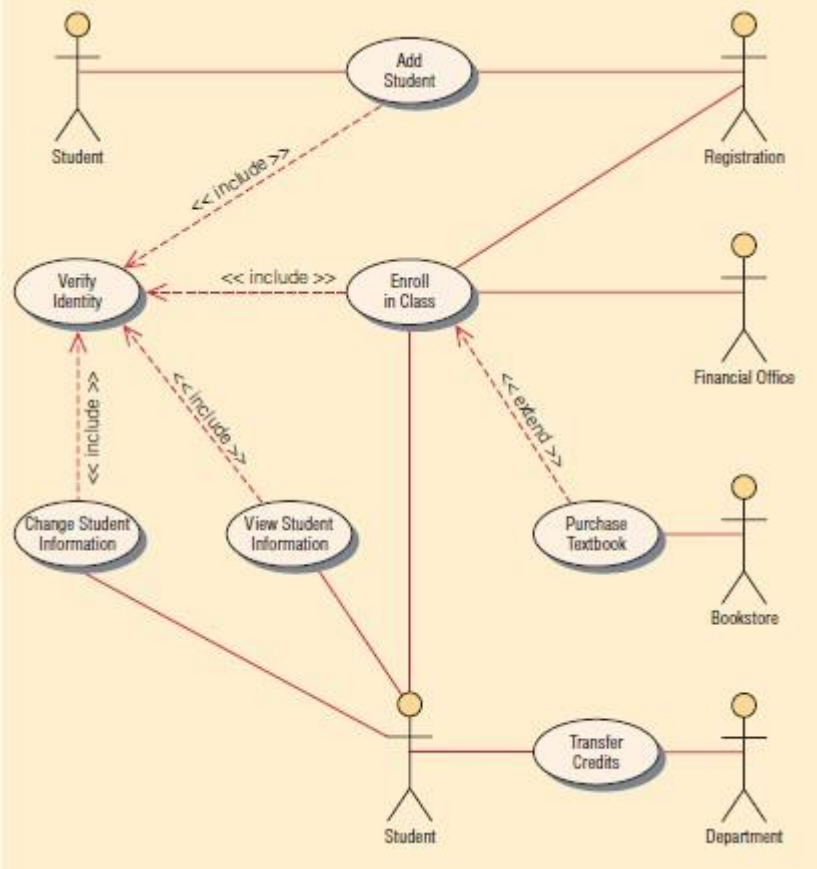
1. *Structure diagrams* yaitu kumpulan diagram yang digunakan untuk menggambarkan suatu struktur statis dari sistem yang dimodelkan.
2. *Behavior diagrams* yaitu kumpulan diagram yang digunakan untuk menggambarkan kelakuan sistem atau rangkaian perubahan yang terjadi pada sebuah sistem.
3. *Interaction diagrams* yaitu kumpulan diagram yang digunakan untuk menggambarkan interaksi sistem dengan sistem lain maupun interaksi antar subsistem pada suatu sistem.

2.21.4 Use Case Digaram

Use case atau diagram *Use case* merupakan pemodelan untuk kelakuan (*behavior*) sistem informasi yang akan dibuat [13]. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. Secara kasar, *Use case* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sebuah sistem informasi dan siapa saja yang berhak menggunakan fungsi- fungsi itu.

Syarat penamaan pada Use case adalah nama didefinisikan sederhana mungkin dan dapat dipahami. Ada dua hal utama pada Use case yaitu

pendefinisian apa yang disebut aktor dan Use case. Contoh Use Case Diagram dapat dilihat pada gambar 2.8.



Gambar 2.8 Contoh Gambar Use Case Diagram

2.21.5 Actor

Actor adalah sesuatu (entitas) yang berhubungan dengan sistem dan berpartisipasi dalam *use case*. *Actor* menggambarkan orang, sistem atau entitas *Eksternal* yang secara khusus membangkitkan sistem dengan *input* atau masukan kejadian-kejadian, atau menerima sesuatu dari sistem. *Actor* dilukiskan dengan peran yang mereka mainkan dalam *use case*, seperti *Staff*, Kurir dan lain-lain.

Dalam *use case* diagram terdapat satu aktor pemulai atau initiator *actor* yang membangkitkan rangsangan awal terhadap sistem, dan mungkin sejumlah aktor lain yang berpartisipasi atau participating *actor*. Akan sangat berguna untuk

mengetahui siapa aktor pemulai tersebut. Bentuk *actor* dapat dilihat pada gambar 2.9.

<<nama actor>>

Gambar 2.9 Bentuk Actor dalam UML

2.21.6 Usecase

Use case yang dibuat berdasarkan keperluan aktor merupakan gambaran dari "apa" yang dikerjakan oleh sistem, bukan "bagaimana" sistem mengerjakannya. *Use case* diberi nama yang menyatakan apa hal yang dicapai dari interaksinya dengan aktor. Bentuk use case dapat dilihat pada gambar 2.10.

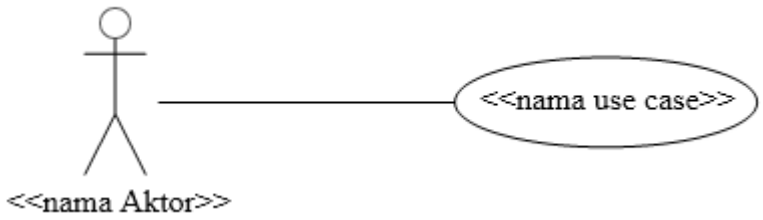
<<nama use case>>

Gambar 2.10 Bentuk Use Case dalam UML

2.21.7 Relationship

Relasi (*relationship*) digambarkan sebagai bentuk garis antara dua simbol dalam *use case diagram*. Relasi antara *actor* dan *use case* disebut juga dengan asosiasi (*association*). Asosiasi ini digunakan untuk menggambarkan bagaimana hubungan antara keduanya.

Relasi-relasi yang terjadi pada *use case diagram* bisa antara *actor* dengan *use case* atau *use case* dengan *use case*. Bentuk relationship dapat dilihat pada gambar 2.11.



Gambar 2.11 Bentuk Relationship dalam UML

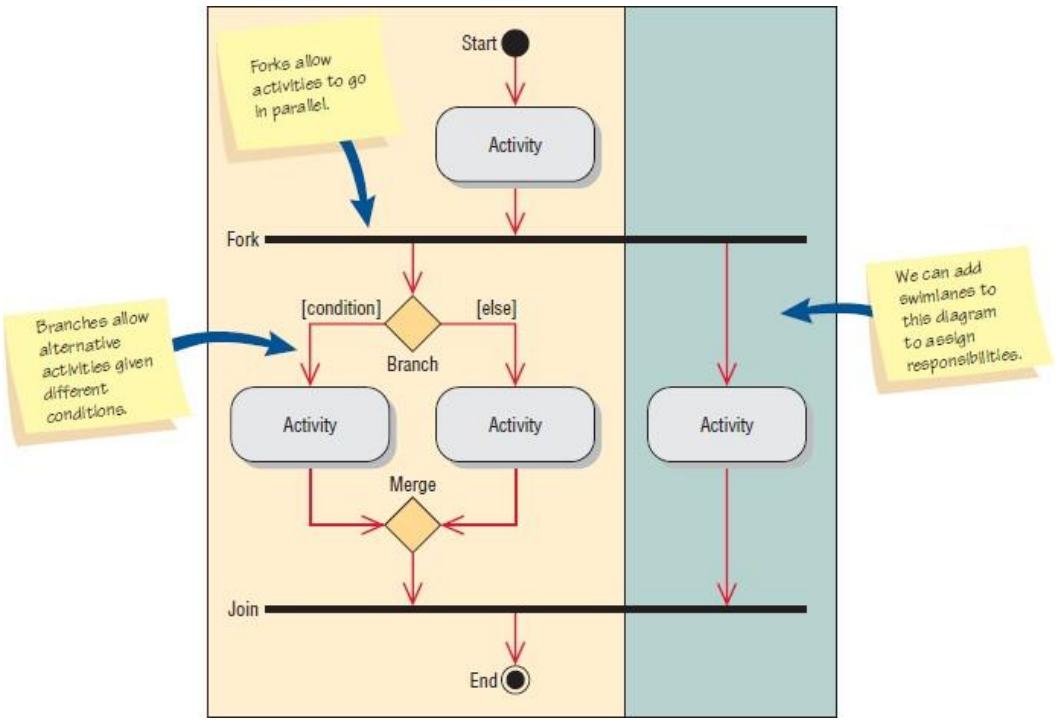
Relasi antara use case dengan use case :

1. *Include,* pemanggilan use case oleh use case lain atau untuk menggambarkan suatu use case termasuk di dalam use case lain (diharuskan). Contohnya adalah pemanggilan sebuah fungsi program. Digambarkan dengan garis lurus berpanah dengan tulisan <<include>>.
2. *Extend,* digunakan ketika hendak menggambarkan variasi pada kondisi perilaku normal dan menggunakan lebih banyak kontrol form dan mendeklarasikan extension pada use case utama. Atau dengan kata lain adalah perluasan dari use case lain jika syarat atau kondisi terpenuhi. Digambarkan dengan garis berpanah dengan tulisan <<extend>>.
3. *Generalization/Inheritance,* dibuat ketika ada sebuah kejadian yang lain sendiri atau perlakuan khusus dan merupakan pola berhubungan base-parent use case. Digambarkan dengan garis berpanah tertutup dari base use case ke parent use case.

2.21.8 Activity Diagram

Diagram aktivitas atau *activity diagram* menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis, yang perlu diperhatikan disini adalah bahwa diagram aktivitas menggambarkan aktivitas sistem bukan apa yang dilakukan aktor, jadi aktivitas yang dapat dilakukan oleh sistem. Diagram aktivitas juga banyak digunakan untuk mendefinisikan hal-hal berikut.

1. Rancangan proses bisnis dimana setiap urutan aktivitas yang digambarkan merupakan proses bisnis sistem yang didefinisikan.
2. Urutan atau pengelompokan tampilan dari sistem/ *user interface* dimana setiap aktivitas dianggap memiliki sebuah rancangan antarmuka tampilan.
3. Rancangan pengujian dimana setiap aktivitas dianggap memerlukan sebuah pengujian yang perlu didefinisikan kasus ujinya [22]. Contoh activity diagram dapat dilihat pada gambar 2.12.



Gambar 2.12 Contoh Activity Diagram

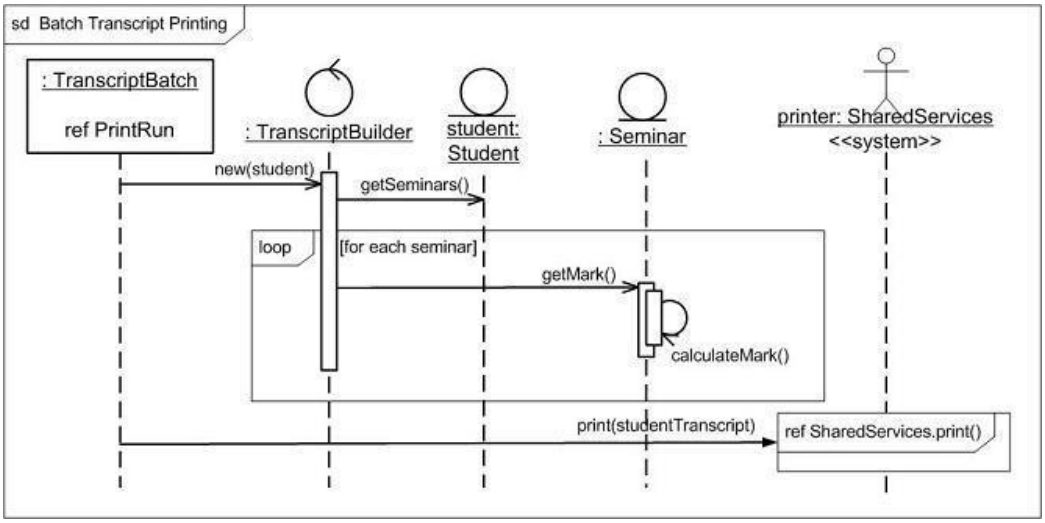
2.21.9 Sequence Diagram

Diagram *sequence* menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan *message* yang dikirimkan dan diterima antar objek. Oleh karena itu untuk menggambarkan diagram *sequence* maka harus diketahui objek-objek yang terlibat dalam sebuah *use case* beserta metode-metode yang dimiliki kelas yang diinstansiasi menjadi objek itu [13].

Diagram *sequence* menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan *message* yang dikirimkan dan diterima

antar objek. Oleh karena itu untuk menggambarkan diagram *sequence* maka harus diketahui objek-objek yang terlibat dalam sebuah *use case* beserta metode-metode yang dimiliki kelas yang diinstansiasi menjadi objek itu.

Penomoran pesan berdasarkan urutan interaksi pesan. Penggambaran letak pesan harus berurutan, pesan yang lebih atas dari lainnya adalah pesan yang berjalan terlebih dahulu. Contoh sequence diagram dapat dilihat pada gambar 2.13.



Gambar 2.13 Contoh Sequence Diagram

2.21.10 Class Diagram

Diagram kelas atau *class diagram* menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. Kelas memiliki apa yang disebut atribut dan metode atau operasi. Atribut merupakan variabel-variabel yang dimiliki oleh suatu kelas. Operasi atau metode adalah fungsi-fungsi yang dimiliki oleh suatu kelas.

Kelas-kelas yang ada pada struktur sistem harus dapat melakukan fungsi-fungsi sesuai dengan kebutuhan sistem. Susunan struktur kelas yang baik pada diagram kelas ebaiknya memiliki jenis-jenis kelas berikut

1. Kelas Main

   Kelas yang memiliki fungsi awal dieksekusi ketika sistem dijalankan.
2. Kelas yang menangani tampilan sistem

   Kelas yang mendefinisikan dan mengatur tampilan ke pemakai.

3. Kelas yang diambil dari pendefinisian usecase

   Kelas yang menangani fungsi-fungsi yang harus ada diambil dari pendefinisian *use case.*

4. Kelas yang diambil dari pendefinisian data

   Kelas yang digunakan untuk memegang atau membungkus data menjadi sebuah kesatuan yang diambil maupun akan disimpan ke basis data.

Dalam mendefinisikan metode yang ada di dalam kelas perlu memperhatikan apa yang disebut dengan *cohesion* dan coupling. *Cohesion* adalah ukuran seberapa dekat keterkaitan instruksi di dalam sebuah metode terkait satu sama lain sedangkan *coupling* adalah ukuran seberapa dekat keterkaitan instruksi antara metode yang satu dengan metode yang lain dalam sebuah kelas. Sebagai aturan secara umum maka sebuah metode yang dibuat harus memiliki kadar *choesion* yang kuat dan kadar *coupling* yang lemah. Dalam *class* diagram terdapat beberapa relasi (hubungan antar *class*) yaitu :

1. *Generalization* dan *Inheritence*

   Diperlukan untuk memperlihatkan hubungan pewarisan (*inheritance*) antar unsur dalam diagram kelas. Suatu hubungan turunan dengan mengasumsikan satu kelas merupakan suatu superClass (kelas super) dari kelas yang lain. Generalization memiliki tingkatan yang berpusat pada superClass.

2. *Associations*

   Suatu hubungan antara bagian dari dua kelas. Terjadi association antara dua kelas jika salah satu bagian dari kelas mengetahui yang lainnya dalam melakukan suatu kegiatan. Di dalam diagram, sebuah association adalah penghubung yang menghubungkan dua kelas.

3. *Aggregation*

   Hubungan antar-*class* dimana *class* yang satu *(part class)* adalah bagian dari *class* lainnya *(whole class).*

4. *Composition*

   Aggregation dengan ikatan yang lebih kuat. Di dalam *composite aggregation*, siklus hidup *part class* sangat bergantung pada *whole class*

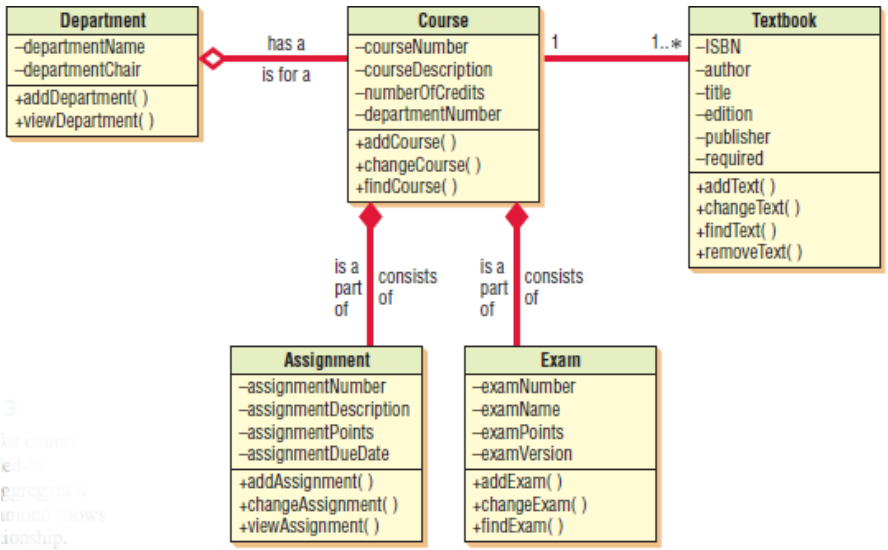
sehingga bila objek *instance* dari *whole class* dihapus maka objek *instance* dari *part class* juga akan terhapus.

5. *Depedency*

   Hubungan antar *class* dimana sebuah *class* memiliki ketergantungan pada *class* lainnya tetapi tidak sebaliknya.

6. *Realization*

   Hubungan antar-*class* dimana sebuah *class* memiliki keharusan untuk mengikuti aturan yang ditetapkan *class* lainnya. Biasanya realization digunakan untk menspesifikasikan hubungan antara sebuah *interface* dengan *class* yang mengimplementasikan *interface* tersebut [22]. Contoh *class* diagram dapat dilihat pada gambar 2.14.



Gambar 2.14 Contoh Class Diagram