BAB 2
TINJAUAN PUSTAKA

2.1 Lari

Lari sangat berbeda dengan berjalan, ketika berlari seorang atlet menggunakan satu kaki yang menyentuh tanah dalam satu waktu, tetapi dalam berlari atau berjalan atlet harus tetap menjaga keseimbangan dan posisi tubuh yang benar, lari jarak pendek dan jarak jauh pada dasarnya memiliki kesamaan dasar, tetapi dalam lari jarak jauh kerja lutut dan lengan bekerja lebih sedikit [6]. Berikut ini akan dijelaskan bagaimana teknik berlari yang baik dan benar.

a. Lari Cepat
    1. Lakukan posisi berdiri
        - Berdiri dibelakang garis start dengan kekuatan pada kaki depan.
        - Tempatkan jari kaki belakang dengan jarak 6-12 dari tumit kaki depan.
        - Tekuk lutut depan hingga 120 derajat dan tempatkan berat pada kaki depan.
        - Tempatkan lengan di depan tumit berlawanan dengan kaki depan dan tubuh diturunkan, serta lengan yang lain dikebelakangkan (misalnya, kaki kiri dan lengan kanan ke depan).
        - Pandanglah dengan ringan dan fokuskan ke depan 2 meter.
    2. Lakukan start berdiri
        - Berdiri dibelakang garis start.
        - Mendengar pistol meletus, majukan kaki ke belakang dengan lutut ke depan kebelakang.
        - Dorong dengan kuat jari kaki dengan kekuatan kaki sambil mengayun lengan belakang ke depan dengan kuat.
        - Merunduk dan gunakan lengan untuk mendorong tubuh kedepan.
    3. Tunjukkan kemampuan start dan finish pada pertandingan lari sprint.
        - Lakukan posisi berdiri.
        - Lakukan start dengan suara pistol atau aba-aba.

- Mintalah untuk tetap dijalur lintasan.
- Pertahankan postur tegak selama sprint.
- Larilah sampai garis finish.

4. Tunjukkan lari sprint yang tepat
   - Lakukan sprint di dalam jalur lintasan.
   - Angkat tumit sehingga sudut pinggul kira-kira mencapai 120 derajat.
   - Ayunkan lengan dengan kuat.
   - Fokuskan pandangan ke lintasan sejauh 10-15 meter.
   - Angkat kepala, dan tegakkan tubuh dengan berpusat pada pinggul.
   - Pertahankan lari sprint yang tepat.

b. Lari jarak jauh

1. Lakukan posisi start untuk lari jarak jauh
   - Tetapkan garis start.
   - Berdiri di belakang garis start dengan salah satu kaki di depan dengan posisi langkah yang ringan.
   - Aturlah berat badan merata di kedua kaki dan bengkokkan lutut sedikit.
   - Usahakan tubuh bagian atas tegak dengan kecondongan ke depan sedikit.
   - Lengan kebawah dan santai di depan tubuh.
   - Fokuslah pandangan 10-15 meter ke lintasan.
2. Lari berdasarkan atas aba-aba start
   - Tetapkan garis start.
   - Lakukan posisi start yang nyaman di belakang garis start.
   - Mulailah start dengan suara pistol (atau aba-aba).
3. Tunjukkan bentuk lari jarak jauh yang tepat
   - Lari dengan langkah sedang.
   - Angkat lutut jangan terlalu tinggi.
   - Tegakkan tubuh dengan sedikit condong ke depan.
   - Tahan kepala dan condong sedikit ke depan.

2.2 *Heart Rate*

Heart Rate (denyut nadi) merupakan jumlah denyut jantung dalam satu menit dengan satuan *Beat Per Minute* (bpm) [7]. Tingkatan denyut nadi akan rendah jika sedang tidak beraktifitas atau istirahat, dan menjadi tinggi jika sedang beraktifitas tinggi, seperti olahraga. Penyebabnya adalah banyaknya darah kaya oksigen yang dibutuhkan oleh tubuh ketika berolahraga. Nilai *heart rate* akan membantu dalam mengevaluasi program latihan saat berolahraga, hal ini juga dapat diterapkan untuk evaluasi dini kesehatan jantung.

Nilai *heart rate* saat istirahat untuk anak-anak sebesar 70-100 bpm untuk usia 6-15 tahun, sedangkan untuk dewasa dengan usia di atas 18 tahun sebesar 60-100 bpm. Nilai spesifik *heart rate* ditunjukka pada tabel sebagai berikut.

Tabel 2.1 Normal *Heart Rate* Berdasarkan Usia

Usia	***Heart Rate* saat terjaga (bpm)**	***Heart Rate* saat tidur (bpm)**
Neonatus (< 28 hari)	100-205	90-160
Bayi	100-190	90-160
Balita (1-2 tahun)	98-140	80-120
Pra-sekolah (3-5 tahun)	80-120	65-100
Sekolah (6-11 tahun)	75-118	58-90
Remaja (12-15 tahun)	60-100	50-90

Detak jantung rata-rata sangat ditentukan oleh keseimbangan antara tingkat aktivitas di saraf simpatetik dan parasimpatik jantung. Penurunan aktivitas vagal dapat dipengaruhi oleh penurunan aktivitas simpatik di titik tertentu, walaupun keduanya otonom. Pada prinsipnya, banyak kombinasi tingkat simpatik dan aktivitas parasimpatik akan menghasilkan hasil yang sama dalam rata-rata denyut jantung.

Detak jantung maksimum adalah detak jantung tertinggi yang dicapai selama latihan. Nilai maksimal dapat ditentukan dengan menggunakan perhitungan untuk pria dan wanita sebagai berikut :

1. Perhitungan HR *Max* Pria
   HR *Max* (Pria) = 206.9 – (0.67 x Usia)
2. Perhitungan HR *Max* Wanita
   HR *Max* (Wanita) = 206 – (0.88 x Usia)

2.3 Saturasi Oksigen

Saturasi oksigen adalah presentasi hemoglobin yang berkaitan dengan oksigen dalam arteri, saturasi oksigen normal adalah antara 95 – 100 % [8]. Pada tekanan parsial oksigen yang rendah, sebagian besar hemoglobin terdeoksigenasi, maksudnya adalah proses pendistribusian darah beroksigen dari arteri ke jaringan tubuh. Pada sekitar 90% (nilai bervariasi sesuai dengan konteks klinis) saturasi oksigen meningkat menurut kurva disosiasi hemoglobin-oksigen dan pendekatan 100% pada tekanan parsial oksigen > 10 kPa. Saturasi oksigen atau oksigen terlarut (DO) adalah ukuran relatif dari jumlah oksigen yang terlarut atau dibawa dalam media tertentu. Hal ini dapat diukur dengan *probe* oksigen terlarut seperti sensor oksigen atau optode dalam media cair. Adapun cara pengukuran saturasi oksigen sebagai berikut.

a. Saturasi oksigen arteri (SaO2) nilai dibawah 90% menunjukkan keadaan hipoksemia (yang juga dapat disebabkan oleh anemia). Hipoksemia karena SaO2 rendah ditandai dengan sianosis. *Oksimetri* nadi adalah metode pemantauan *non invasif* secara kontinyu terhadap saturasi oksigen hemoglobin (SaO2).

   Meski *oksimetri* oksigen tidak bisa menggantikan gas-gas darah arteri, *oksimetri* oksigen merupakan salah satu cara efektif untuk memantau pasien terhadap perubahan saturasi oksigen yang kecil dan mendadak. *Oksimetri* nadi digunakan dalam banyak lingkungan, termasuk unit perawatan kritis, unit keperawatan umum, dan pada area diagnostik dan

pengobatan ketika diperlukan pemantauan saturasi oksigen selama prosedur.

b. Saturasi oksigen vena (SvO2) diukur untuk melihat berapa banyak mengkonsumsi oksigen tubuh. Dalam perawatan klinis, SvO2 di bawah 60% menunjukkan bahwa tubuh dalam kekurangan oksigen. Pengukuran ini sering digunakan pengobatan dengan mesin jantung-paru (*Extracorporeal Sirkulasi*), dan dapat memberikan gambaran tentang berapa banyak aliran darah pasien yang diperlukan agar tetap sehat.
c. *Tissue* oksigen saturasi (StO2) dapat diukur dengan spektroskopi inframerah dekat. *Tissue* oksigen saturasi memberikan gambaran tentang oksigenasi jaringan dalam berbagai kondisi.
d. Saturasi oksigen perifer (SpO2) adalah estimasi dari tingkat kejenuhan oksigen yang biasanya diukur dengan oksimeter pulsa.

2.4 Internet of Things (IoT)

Internet of Things (IoT) merupakan struktur dimana objek, orang disediakan dengan identitas eksklusif dan kemampuan untuk pindah data melalui jaringan tanpa memerlukan dua arah antara manusia ke manusia yaitu sumber ke tujuan atau interaksi manusia ke computer.

Internet of things merupakan perkembangan keilmuan yang sangat menjajikan untuk mengoptimalkan kehidupan berdasarkan sensor cerdas dan peralatan pintar yang berkerjasama melalui jaringan internet [9].

2.5 Mikrokontroler

Mikrokontroler merupakan suatu kontroler yang digunakan untuk mengontrol suatu proses atau aspek-aspek dari lingkungan. Satu contoh aplikasi dari mikrokontroler adalah untuk memonitor rumah kita. Ketika suhu naik kontroler membuka jendela dan sebaliknya. Pada masanya, kontroler dibangun dari komponen-komponen logika secara keseluruhan, sehingga menjadikannya besar dan berat. Setelah itu barulah dipergunakan mikroprosesor sehingga keseluruhan kontroler masuk kedalam PCV yang cukup kecil. Hingga saat ini masih sering kita

lihat kontroler yang dikendalikan oleh mikroprosesor biasa (Zilog Z80, Intel 8088, Motorola 6809, dsb).

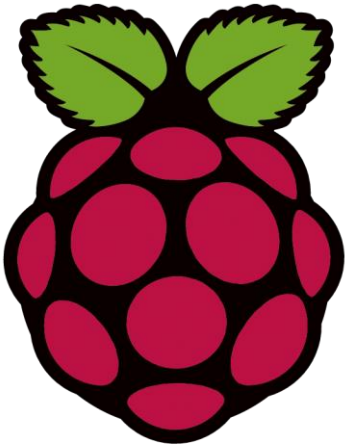
Prose pengecilan komponen terus berlangsung, semua komponen diperlukan guna membangun suatu kontroler dapat dikemas dalam satu keping. Maka lahirlah komputer keping tunggal (one chip microcomputer) atau disebut juga mikrokontroler. Mikrokontroler adalah suatu IC dengan kepadatan yang sangat tinggi, dimana semua bagan yang diperlukan untuk suatu kontroler sudah dikemas dalam satu keping, biasanya terdiri dari :

1. CPU (Central Processing Unit)
2. RAM (Random Access Memory)
3. EEPROM/EPROM/PROM/ROM
4. I/O, Serial dan Paralel
5. Timer
6. Interupt Controller.

Rata-rata mikrokontroler memiliki intruksi manipulasi bit, akses ke I/O secara langsung dan mudah, dan proses *interrupt* yang cepat dan efisien. Dengan kata lain mikrokontroler adalah “Solusi satu Chip” yang secara drastis mengurangi jumlah komponen dan biaya desain [10].

2.6 Raspberry Pi

Raspberry Pi, sering disingkat dengan nama Raspi, adalah komputer papan tunggal/*single-board circuit* (SBC) yang seukuran dengan kartu kredit yang dapat digunakan untuk menjalankan program perkantoran, permainan komputer, dan sebagai pemutar media hingga video beresousi tinggi. Raspberry Pi dikembangkan oleh yayasan nirlaba, Rasberry Pi Foundation, yang digawangi sejumlah pengembang dan ahli komputer dari Universitas Cambridge, Inggris.



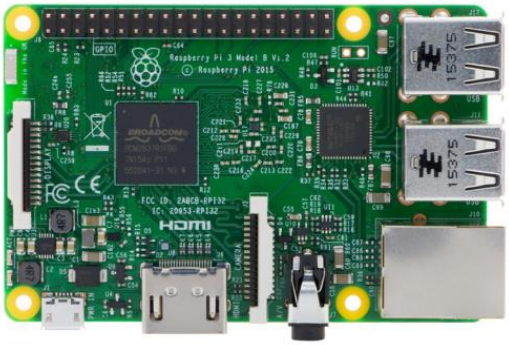
Gambar 2.1 Lambang Raspberry Pi

Ide dibalik Raspberry Pi diawali dari keinginan untuk mencetak pemrogram generasi baru. Seperti disebutkan dalam situs resmi Raspberry Pi Foundation, waktu itu Eben Upton, Rob Mullins, Jack Lang, dan Alan Mycroft, dari Laboratorium Komputer Universitas Cambridge memiliki kekhawatiran melihat kian turunnya keahlian dan jumlah siswa yang hendak belajar ilmu komputer. Mereka lantas mendirikan yayasan Raspberry Pi bersama dengan Pete Lomas dan David Braben pada 2009. Tiga tahun kemudian, Raspberry Pi Model B memasuki produksi massal. Dalam peluncuran pertamanya pada akhir Febuari 2012 dalam beberapa jam saja sudah terjual 100.000 unit. Pada bulan Februari 2016, Raspberry Pi Foundation mengumumkan bahwa mereka telah menjual 8 juta perangkat Raspi, sehingga menjadikannya sebagai perangkat paling laris di Inggris [11].

2.7 Raspberry Pi 3 Model B

Raspberry Pi 3 Model B merupakan generasi ketiga dari keluarga raspberry pi. raspberry pi 3 memiliki RAM 1GB dan grafis Broadcom VideoCore IV pada frekuensi clock yang lebih tinggi dari sebelumnya yang berjalan pada 250MHz. Raspberry Pi 3 juga memiliki 4 USB port, 40 pin GPIO, Full HDMI port, Port Ethernet, Combined 3.5mm *audio jack and composite video*, *Camera interface* (CSI), *Display interface* (DSI), slot kartu Micro SD (Sistem tekan-tarik, berbeda

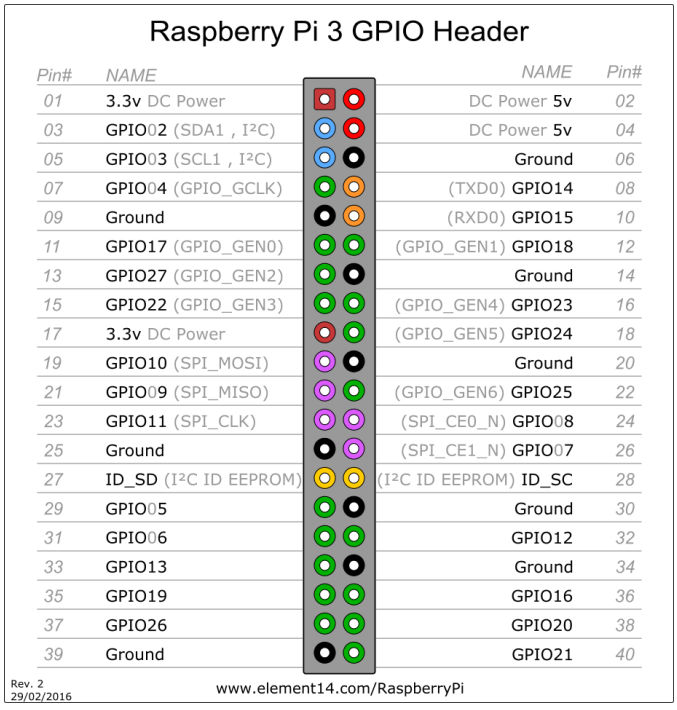
dari yang sebelum nya ditekan-tekan), dan VideoCore IV 3D *graphics core* [12]. Bentuk Raspberry Pi 3 Model B dapat dilihat dalam gambar berikut :



Gambar 2.2 Raspberry Pi 3 Model B

2.7.1 GPIO Raspberry Pi 3 Model B

Raspberry Pi 3 Model B terdapat 40 buah pin. Pin pada raspberry pi 3 model B terdiri dari beberapa bagian yaitu bagian VCC, GND, dan GPIO (*General Purpose Input/Output*) terdapat 3 pin VCC dan 8 pin GND. Pin GPIO mulai dari GPIO 2 hingga GPIO 27 [12]. Pada pin GPIO terdapat fungsi lain yang dapat dilihat pada gambar berikut :



Gambar 2.3 Pin GPIO Raspberry Pi 3 Model B

2.8 Sensor

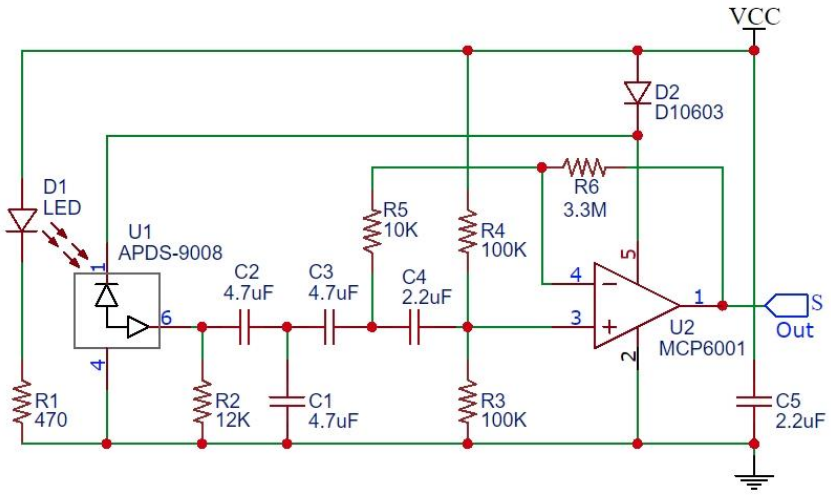
Sensor adalah alat untuk mendeteksi/mengukur sesuatu, yang digunakan untuk mengubah variasi mekanis, magnetis, panas, sinar, dan kimia menjadi tegangan dan arus listrik. Sensor berfungsi mengubah besaran fisik (misalnya : temperature, gaya, kecepatan putaran) menjadi besaran listrik.

2.8.1 Modul *Heart Rate Pulse Sensor*

Modul *Heart Rate Pulse Sensor* pada dasarnya menggunakan prinsip kerja *photoplethysmography*, dimana merupakan metode optis yang relatif sederhana dan murah untuk mendeteksi secara *non-invansive* perubahan volume darah setiap jantung berdetak pada jaringan pembuluh darah [13]. Bentuk fisik *heart rate pulse sensor* dapat dilihat pada gambar berikut :

Gambar 2.4 Bentuk Fisik Modul *Heart Rate Pulse Sensor*

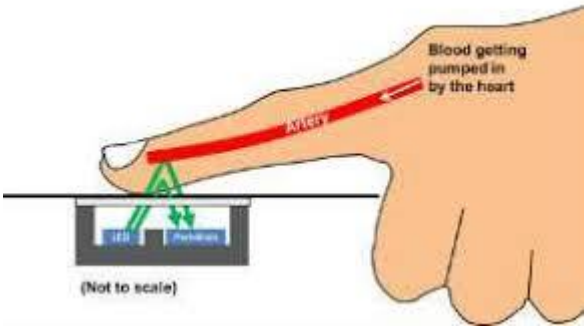
Bagian depan sensor ini akan ditempelkan pada jari tangan untuk mendeteksi detak jantung pasien atau naracoba. Di dalam modul sensor ini telah diintegrasikan sejumlah fungsi komponen yaitu sensor *infrared*, detektor cahaya, rangkaian filter, rangkaian penguat sinyal dan komponen proteksi rangkaian. Skematik diagram sederhana modul ini ditunjukkan pada gambar berikut :



Gambar 2.5 Diagram Skematik *Heart Rate Pulse Sensor*

Sinyal yang dihasilkan oleh sensor menghasilkan gelombang yang dinamakan *photoplethysmogram* (PPG). PPG dalam dunia medis digunakan untuk pengukuran *respiratory rate* (pernafasan) dan *heart rate* (detak jantung). Saat jantung memompa darah ke seluruh tubuh, setiap denyut yang terjadi disertai dengan munculnya gelombang pulsa seperti gelombang kejut yang merambat melalui arteri hingga ke lapisan kapiler tangan (jemari) tempat *heart rate pulse sensor* dipasang.

Photodiode yang sudah terintegrasi dalam komponen APDS 9008 digunakan sebagai penangkap gelombang cahaya yang dipancarkan oleh *Infra Red* (IR). Metoda pengukuran detak jantung pada pembuluh darah jari tangan pada sistem ini menggunakan metoda refleksi, dimana IR sebagai sumber cahaya dipasangkan sejajar dengan *Photodiode* sebagai sensor cahaya. Sinyal atau perubahan yang diterima oleh *photodiode* adalah pantulan cahaya dari IR. *Photodiode* mengubah besarnya intensitas cahaya yang diterima menjadi arus listrik. Besar kecilnya cahaya yang diterima berdasarkan pantulan cahaya dari IR yang dipancarkan ke pembuluh darah pada jari tangan.



Gambar 2.6 Metode Refleksi Untuk Memantau Detak Jantung

Arus listrik yang dihasilkan oleh komponen APDS-9008 kemudian diubah menjadi tegangan listrik di titik TP0 dengan menggunakan sebuah resistor 12kΩ. tegangan listrik ini kemudian disaring untuk menghilangkan tegangan DC dan diperkecil tegangannya dengan menggunakan kapasitor C1 dan C2. Sinyal ini kemudian dihubungkan rangkaian *differensiator op-amp* dengan frekuensi *cut-off* 3,38 Hz yang dihasilkan oleh kapasitor C3 dan R5. Rangkaian *differensiator* ini akan berfungsi ganda yaitu sebagai *differensiator* jika frekuensi sinyal *input* pada TP1 lebih kecil dari frekuensi *cut-off* atau rangkaian akan berfungsi sebagai penguat *inverting* jika frekuensi sinyal input pada TP1 lebih besar dari frekuensi *cut-off*. Sinyal pada TP1 kemudian dihubungkan dengan komponen C4 dan resistor R3 untuk difilter kembali dengan frekuensi *cut-off* 0,72 Hz. Dengan kata lain, rangkaian ini akan memfilter sinyal sesuai dengan detak jantung manusia pada umumnya yaitu lebih kurang 43 sampai dengan 200 beat per minute (bpm).

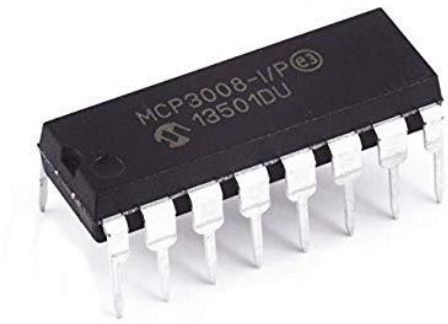
Komponen resistor R3 dan R4 berfungsi juga sebagai pembagi tegangan untuk menghasilkan tegangan bias untuk sinyal TP1 sehingga memiliki sinyal dengan tegangan *offset* sebesar setengah dari tegangan catu daya (Vcc) yang diberikan. Sinyal tersebut kemudian dikondisikan dengan rangkaian *differensiator op-amp* sesuai dengan frekuensi sinyal input yang mengalir pada rangkaian. Jika rangkaian *differensiator* ini berfungsi sebagai penguat, maka akan menghasilkan penguat *inverting* dengan penguatan sebesar 330 kali. Tegangan keluaran (Vout) dari modul ini berupa level tegangan DC yang memenuhi persyaratan untuk diproses lebih

lanjut oleh modul pemroses sinyal ADC lalu dikirimkan ke mikrokontroler Raspberry Pi.

Modul sensor ini juga dilengkapi dengan pengaman berupa diode D2 untuk mencegah terbaliknya polaritas catu daya yang diberikan agar menghindari kerusakan pada komponen. Diode D2 akan bersifat *forward* bias jika polaritas catu daya dalam posisi yang salah.

2.9 Analog to Digital Converter MCP3008

MCP3008 adalah 8-*channel* 10 bit *analog-to-digital converter* (ADC) dengan paket 16-pin PDIP dan SOIC. MCP3008 dapat diprogram untuk menyediakan empat pasangan *input pseudo-differensial* atau delapan *single-ended input*. *Differensial Non-Linearity* (DNL) dan *Integral Non-Linearity* (INL) ditetapkan pada ±1 LSB. Komunikasi dengan perangkat dilakukan dengan menggunakan antarmuka serial sederhana yang kompatibel dengan protokol SPI. Perangkat ini mampu tingkat konversi hingga 200 ksps. Perangkat MCP3008 beroperasi pada rentang tegangan (2.7V – 5.5V). Desain arus rendah ini memungkinkan pengoperasian dengan arus siaga tipikal hanya 5 nA dan arus aktif tipikal 320 µA [14].



Gambar 2.7 Analog to Digital Converter MCP3008

2.10 *Object Oriented (OO)*

Object Oriented adalah sebuah obyek memiliki keadaan sesaat (*state*) dan perilaku (*behaviour*). State sebuah obyek adalah kondisi obyek tersebut yang

dinyatakan dalam *attribute/properties*. *State* sebuah obyek adalah kondisi obyek tersebut yang dinyatakan dalam attribute/properties. Sedangkan perilaku suatu obyek mendefinisikan bagaimana sebuah obyek bertindak/beraksi dan memberikan reaksi.

Perilaku sebuah objek dinyatakan dalam operation. Menurut Schmuller, *attribute* dan *operation* bila disatukan akan memberikan fitur/*features*. Himpunan objek-objek yang sejenis disebut class. Objek adalah contoh *instance* dari sebuah *class*. Ada dua macam aplikasi yang tidak cocok dikembangkan dengan metode OO. Yang pertama adalah aplikasi yang sangat berorientasi ke *database*. Aplikasi yang sangat berorientasi ke penyimpanan dan pemanggilan data sagnat tidak cocok dikembangkan dengan OO karena akan banyak kehilangan manfaat dari penggunaan RDBMS (*Relational Database Management System*) untuk penyimpanan data. Akan tetapi RDBMS juga mempunyai keterbatasan dalam penyimpanan dan pemanggilan struktur data yang kompleks seperti multimedia, data spasial dan lain-lain. Bentuk aplikasi lain yang kurang cocok dengan pendekatan OO adalah aplikasi yang membutuhkan banyak algoritma.

Beberapa applikasi yang melibatkan perhitungan yang besar dan kompleks. Pada penelitian ini konsep OO digunakan untuk konsep pengkodean pada sistem yang akan dibangun [15]. Berikut ini adalah konsep-konsep dalam pemrograman berorientasi objek :

1. ***Class***

   Kelas (*Class*) merupakan penggambaran satu set objek yang memiliki atribut yang sama. Kelas mirip dengan tipe data ada pemrograman non objek, akan tetapi lebih komprehensif karena terdapat struktur sekaligus karakteristiknya. Kelas baru dapat dibentuk lebih spesifik dari kelas ada umumnya.kelas merupakan jantung dalam pemrograman berorientasi objek.

2. ***Object***

   Objek merupakan teknik dalam menyelesaikan masalah yang kerap muncul dalam pengembangan perangkat lunak. Teknik ini merupakan teknik yang efektif dalam menemukan cara yang tepat dalam membangun sistem dan

menjadi metode yang paling banyak dipakai oleh para pengembang perangkat lunak. Orientasi objek merupakan teknik pemodelan sistem riil yang berbasis objek.

3. ***Abstraction***

   Kemampuan sebuah program untuk melewati aspek informasi yang diolah adalah kemampuan untuk fokus pada inti permasalahan. Setiap objek dalam sistem melayani berbagai model dari pelaku abstrak yang dapat melakukan kerja, laporan dan perubahan serta berkomunikasi dengan objek lain dalam sistem, tanpa harus menampakkan kelebihan diterapkan.

4. ***Encapsulation***

   Encapsulation adalah proses memastikan pengguna sebuah objek tidak dapat menggantikan keadaan dari sebuah objek dengan cara yang tidak sesuai prosedur. Artinya, hanya metode yang terdapat dalam objek tersebut yang diberi izin untuk mengakses keadaan yang diinginkan. Setiap objek mengakses *interface* yang menyabutkan bagaimana objek lainnya dapat berintegrasi dengannya. Objek lainnya tidak akan mengetahui dan tergantung kepada representasi dalam objek tersebut.

5. ***Polimorfism***

   Polimorfism merupakan suatu fungsionalitas yang diimplikasikan dengan berbagai cara yang berbeda. Pada program berorientasi objek, pembuat program dapat memiliki berbagai implementasi untuk sebagian fungsi tertentu.

6. ***Inheritance***

   Seperti yang sudah diuraikan di atas, objek adalah contoh / *instance* dari sebuah *class*. Hal ini mempunyai konsekuensi yang penting yaitu sebagai *instance* sebuah *class*, sebuah objek mempuyai semua karakteristik dari classnya. Inilah yang disebut dengan *Inheritance* (pewarisan sifat). Dengan demikian apapun *attribute* dan *operation* dari *class* akan dimiliki pula oleh semua obyek yang disinherit/diturunkan dari class tersebut. Sifat ini tidak hanya berlaku untuk objek terhadap *class*, akan tetapi juga berlaku untuk *class* terhadap *class* lainnya.

2.11 UML (*Unified Modeling Language*)

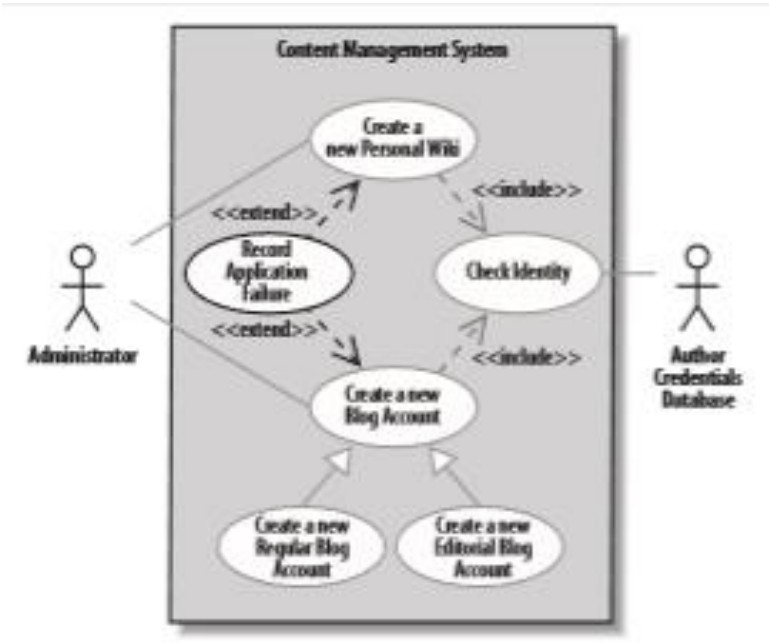
Unified Modelling Language (UML) adalah bahasa pemodelan visual yang digunakan untuk menggambarkan, membangun dan mendokumentasikan alur dari sistem perangkat lunak [16]. Digunakan untuk memahami, mendesain, menelusuri, mengkonfigurasi, memelihara, dan mengontrol informasi tentang suatu sistem.

Dengan semua metode pengembangan, tahapan siklus hidup, domain aplikasi, dan media, UML berguna bagi developer untuk mengetahui komponen mana yang dibutuhkan, apa yang komponen tersebut lakukan, dan bagaimana mereka cocok dengan keinginan konsumen, dan jika perangkat lunak tersebut dibangun oleh sebuah tim, UML juga membantu agar setiap bagian-bagian yang nantinya akan dibagikan kepada tiap anggota tim lainnya tidak menjadi berantakan.

Berikut adalah beberapa model yang digunakan pada bahasa UML.

1. *Use Case Diagram*

   Use case adalah sebuah situasi dimana sistem yang dibangun digunakan untuk memenuhi salah satu atau lebih dari keinginan pemakainya; sebuah *use case* mengambil bagian dari fungsionalitas yang sistem tersebut berikan. *Use case* terdapat pada jantung dari setiap model. Contoh *use case diagram* dapat dilihat pada Gambar



Gambar 2.8 Diagram Use Case

2. Use Case Scenario

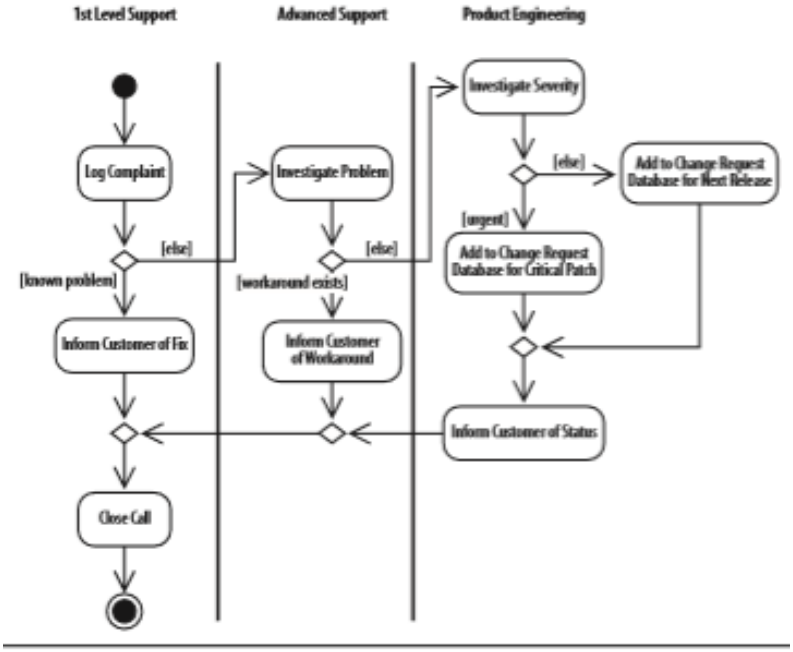
Pada *use case diagram* telah ditunjukkan setiap situasi dan aktor yang terlibat dalam sistem tersebut, tapi *use case* tidak menjelaskan secara detail bagaimana nantinya setiap *case* akan bertemu dan siapa aktor utama yang menggunakan *case* tersebut serta langkah-langkahnya, disinilah *use case scenario* digunakan. Contoh *use case scenario* dapat dilihat pada Gambar

Use case name	**Create a new Blog Account**	
Successful End Condition	A new blog account is created for the author.	
Failed End Condition	The application for a new blog account is rejected.	
Primary Actors	Administrator	
Secondary Actors	None	
Trigger	The Administrator asks the CMS to create a new blog account.	
Included Cases	Check Identity	
Main Flow	**Step**	**Action**
	1	The Administrator asks the system to create a new blog account.
	2	The Administrator selects an account type.
	3	The Administrator enters the author's details.
	4	The author's details are checked.
	include=Check Identity	
	5	The new account is created.
	6	A summary of the new blog account's details are emailed to the author.

Gambar 2.9 Use Case Scenario

3. Activity Diagram

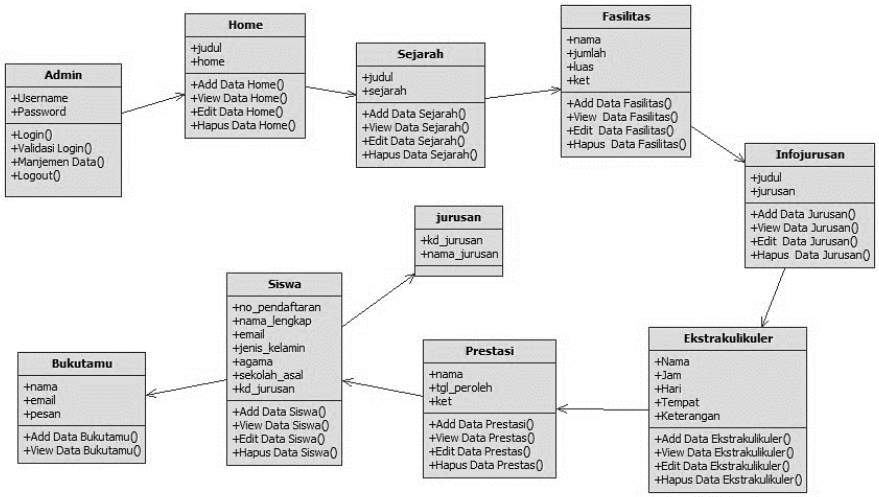
Activity diagram menunjukkan bagaimana sistem tersebut akan mencapai tujuannya. Diagram aktivitas sangat bagus digunakan untuk memodelkan proses bisnis. Contoh dari *activity diagram* dapat dilihat pada gambar



Gambar 2.10 Activity Diagram

4. Class Diagram

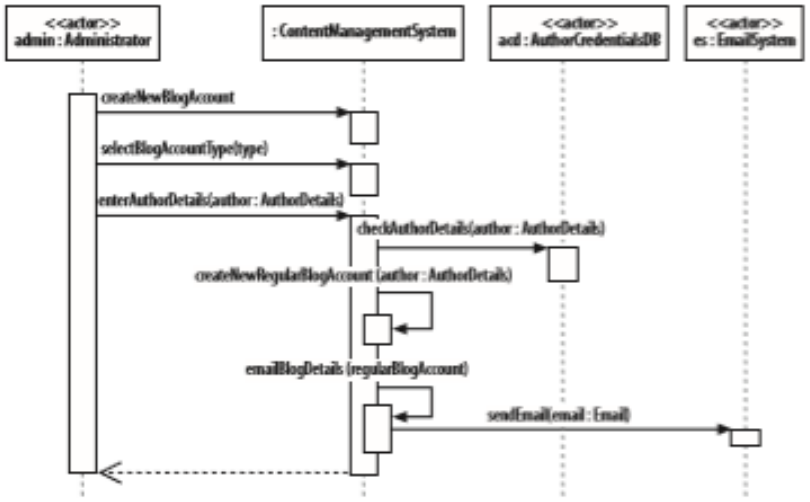
Class Diagram adalah jantung bagi sistem berorientasi objek. Oleh karena itu, hal tersebut diikuti dengan disebutnya *class diagram* sebagai diagram UML yang paling populer. Struktur sebuah sistem terbuat dari kumpulan bagian yang disebut sebagai objek. C*lass* menggambarkan perbedaan tipe-tipe objek yang sistem tersebut miliki, dan diagram kelas menunjukkan kelas-kelas tersebut dan relasinya. Contoh *class diagram* dapat dilihat pada Gambar



Gambar 2.11 Class Diagram

5. Sequence Diagram

Sequence Diagram menggambarkan interaksi alur waktu antar bagian yang membangun sistem tersebut dan bentuk logika dari model itu sendiri. Contoh dari *sequence diagram* dapat dilihat pada Gambar



Gambar 2.12 Sequence Diagram

2.12 Python

Python diciptakan oleh Guido van Rossum di Belanda pada tahun 1990 dan namanya diambil dari acara televisi kesukaan *Guido Monty Python's Flying Circus*. Van Rossum mengembangkan Python sebagai hobi, kemudian Python menjadi bahasa pemrograman yang dipakai secara luas dalam industri dan pendidikan karena sederhana, ringkas, sintaks intuitif dan memiliki pustaka yang luas.

Python saat ini dikembangkan dan dikelola oleh tim relawan yang besar dan tersedia secara gratis dari *Python Software Foundation*. Python termasuk dari jajaran bahasa pemograman tingkat tinggi seperti bahasa pemograman C, C++, Java, Perl dan Pascal, dikenal juga bahasa pemograman tingkat rendah, yang dikenal sebagai bahasa mesin yaitu bahasa pemograman *assembly*, kenyataanya komputer hanya dapat mengeksekusi bahasa tingkat rendah, jadi bahasa tingkat tingkat tinggi harus melewati beberapa proses untuk diubah ke bahasa pemograman tingkat rendah, hal tersebut merupakan kelemahan yang tidak berarti bahasa

pemograman tingkat tinggi. Tetapi kekurangan tersebut tidak sebanding dengan kelebihanya.

Pertama, lebih mudah memprogram sebuah aplikasi dengan bahasa tingkat tinggi. Lebih cepat, lebih mudah dimengerti menulis program komputer dengan bahasa tingkat tinggi, dan juga kesalahan dalam penulisan program cenderung tidak mengalami kesalahan yang berarti. Kedua bahasa pemograman tingkat tinggi lebih portabel dalam arti bisa digunakan untuk menulis di berbagai jenis arsitektur komputer yang berlainan dengan sedikit modifikasi ataupun tidak memerlukan modifikasi sama sekali.

Bahasa pemograman tingkat rendah hanya dapat berjalan di satu jenis arsitektur komputer dan harus ditulis ulang untuk menjalankannya di lain mesin, hal ini dikarenakan karena perbedan urutan *register* dan *services* – *services*nya. Pada penelitian ini bahasa pemograman python digunakan sebagai komponen utama untuk pembanguan *website* dan konfigurasi raspberry pi [17].

Gambar 2.13 Logo Python

Python digunakan di berbagai bidang pengembangan. Berikut beberapa implementasi bahasa python yang paling populer :

1. Website

   Bahasa pemograman python dapat digunakan sebagai server side yang diintegrasikan dengan berbagai internet *protocol* misalnya JSON, *Email Processing*, FTP, dan IMAP. Selain itu python juga mempunyai library pendukung untuk pengembangan website seperti Django, Flask, Pyramid dan Bottle.

2. Penelitian

   Python dapat digunakan juga untuk melakukan riset ilmiah untuk mempermudah perhitungan numerik. Misalnya penerapan algoritma KNN,

Naïve Bayes dan lain-lain. Beberapa *library* yang sering digunakan untuk riset seperti Pandas, Numpy, Mathplotlib dan lain-lain.

3. Media Pembelajaran

   Python dapat digunakan sebagai media pembelajaran di universitas. Python sangat mudah dan hemat untuk dipelajari sebagai *Object Oriented Programming* dibandingkan bahasa lainnya seperti MATLAB, C++, dan C#.

4. *Graphical User Interface* (GUI)

   Python dapat digunakan untuk membangun *interface* sebuah aplikasi. Tersedia library untuk membuat GUI menggunakan python, misalnya Qt, win32extension, dan GTK+.

5. *Internet of Things* (IoT)

   Bahasa pemrograman Python mendukung ekosistem *Internet of Things* (IOT) dengan sangat baik. *Internet Of Things* (IOT) merupakan sebuah teknologi yang menghubungkan benda-benda di sekitar kita ke dalam sebuah jaring-jaring yang saling terhubung.

2.13 MySQL

MySQL adalah sebuah perangkat lunak sistem manajemen basis data SQL (bahasa Inggris: *database management system*) atau DBMS yang multithread, multi-user, dengan sekitar 6 juta instalasi di seluruh dunia. MySQL AB membuat MySQL tersedia sebagai perangkat lunak gratis di bawah lisensi GNU *General Public License* (GPL), tetapi mereka juga menjual dibawah lisensi komersial untuk kasus-kasus dimana penggunaannya tidak cocok dengan penggunaan GPL. Pada penelitian ini MySQL digunakan sebagai platform penyimpanan data dan implementasi model rancangan database yang sudah di buat.

Gambar 2.14 Logo MySQL

MySQL memiliki banyak fitur. fitur dimiliki MySQL sebagai berikut :

1. *Relational Database System,* Seperti halnya software database lain yang ada di pasaran, MySQL termasuk RDBMS.
2. *Arsitektur Client-Server, MySQL* memiliki arsitektur client-server dimana *server database MySQL* terinstal di *server*. Client *MySQL* dapat berada di komputer yang sama dengan *server*, dan dapat juga di komputer lain yang berkomunikasi dengan server melalui jaringan bahkan *internet.*
3. Mengenal perintah SQL standar, SQL *(Structured Query Language)* merupakan suatu bahasa standar yang berlaku di hampir semua software database. *MySQL* mendukung *SQL* versi *SQL:2003*.
4. *Performace Tuning*, *MySQL* mempunyai kecepatan yang cukup baik dalam menangani *query-query* sederhana, serta mampu memproses lebih banyak *SQL* per satuan waktu.
5. *Sub Select.*
6. *Views.*
7. *Stored Prosedured* (SP).
8. *Triggers.*
9. *Replication.*
10. *Foreign Key.*
11. Security yang baik.

2.14 Black Box Testing

Black box testing merupakan metode pengujian dengan pendekatan yang mengasumsikan sebuah sistem perangkat lunak atau program sebagai sebuah kotak hitam (*black box*). Pendekatan ini hanya mengevaluasi program dari output atau

hasil akhir yang dikeluarkan oleh program tersebut [18]. Struktur program dan kode-kode yang ada di dalamnya tidak termasuk dalam pengujian ini. Keuntungan dari metode pengujian ini adalah mudah dan sederhana. Namun, pengujian dengan metode ini tidak dapat mendeteksi kekurangefektifan pengkodean dalam suatu program. Metode pengujian *black box* merupakan metode pengujian dengan pendekatan yang mengasumsikan sebuah sistem perangkat lunak atau program sebagai sebuah kotak hitam (*black box*). Pendekatan ini hanya mengevaluasi program dari output atau hasil akhir yang dikeluarkan oleh program tersebut. Struktur program dan kode-kode yang ada di dalamnya tidak termasuk dalam pengujian ini. Keuntungan dari metode pengujian ini adalah mudah dan sederhana. Namun, pengujian dengan metode ini tidak dapat mendeteksi kekurangefektifan pengkodean dalam suatu program. Ciri-ciri *black box testing* adalah sebagai berikut:

1. *Black box testing* berfokus pada kebutuhan fungsional pada *software*, berdasarkan pada spesifikasi kebutuhan dari *software*.
2. Merupakan pendekatan pelengkap dalam mencangkup *error* dengan kelas yang berbeda dari metode *white box testing*.
3. Melakukan pengujian tanpa pengetahuan detil struktur internal dari sistem atau komonen yang dites. Juga disebut sebagai *behavioural testing, specification-based testing, input/ouput testing* atau *functional testing*.
4. Terdapat jenis *test* yang dapat dipilih berdasarkan pada tipe testing yang digunakan.
5. Kategori *error* yang akan diketahuai melalui *black box testing* seperti fungsi yang hilang atau tidak benar, *error* dari antar-muka, *error* dari struktur data atau akses eksternal database, *error* dari kinerja dan *error* dari inisialisasi.

Equivalence class partitioning adalah sebuah metode black box terarah yang meningkatkan efisiensi dari pengujian dan meningkatkan *coverage* dari error yang potensial. Sebuah *equivalence class* adalah sebuah kumpulan dari nilai *variable input* yang memproduksi *output* yang sama. Selanjutnya *output correctness test* merupakan pengujian yang memakan sumber daya paling besar dari pengujian.

Pada kasus yang sering terjadi dimana hanya *output correctness test* yang dilakukan, maka sumber daya pengujian akan digunakan semua. Implementasi dari kelas-kelas pengujian lain tergantung dari sifat produk *software* dan pengguna selanjutnya dan juga prosedur dan keputusan pengembang. *Output correctness test* mengaplikasikan konsep dari *test case*. Pemilihan *test case* yang baik dapat dicapai dengan efesiensi dari penggunaan *equivalence class partitioning*. Jenis-jenis *test case* sebagai berikut:

1. *Availability test*

Availability didefinisikan sebagai waktu reaksi yaitu waktu yang dibutuhkan untuk mendapatkan informasi yang diminta atau waktu yang dibutuhkan oleh *firmware* yang diinstal pada perlengkapan komputer untuk bereaksi. *Availability* adalah yang paling penting dalam aplikasi online sistem informasi yang sering digunakan. Kegagalan *firmware software* untuk memenuhi persyaratan ketersediaan dapat membuat perlengkapan tersebut tidak berguna.

2. *Realiability test*

Realiability berkaitan dengan fitur yang dapat diterjemahkan sebagai kegiatan yang terjadi sepanjang waktu seperti waktu rata-rata antara kegagalan (misalnya 500 jam), waktu rata-rata untuk *recovery* setelah kegagalan sistem (misalnya 15 menit) atau *average downtime* per bulan (misalnya 30 menit per bulan). Persyaratan realiabilitas memiliki efek selama *regular full-capacity* operasi sistem. Harus diperhatikan bahwa penambahan faktor *software realiability* juga berkaitan dengan perangkat, sistem operasi, dan efek dari sistem komunikasi data.

3. *Stress test*

Stress test terdiri dari 2 tipe pengujian yaitu *load test* dan *durability test*. Suatu hal yang mungkin untuk melakukan pengujian-pengujian tersebut setelah penyelesaian sistem *software*. *Durability test* dapat dilakukan hanya setelah *firmware* atau sistem informasi *software* diinstall dan siap untuk diuji. Pada *load test* berkaitan dengan *functional performance system* dibawah beban maksimal operasional, yaitu maksimal transaksi per menit, hits per menit ke tempat internet dan sebagainya. *Load test*, yang biasanya dilakukan untuk beban yang lebih tinggi dari yang diindikasikan spesifikasi persyaratan merupakan hal yang penting untuk

sistem *software* yang rencananya akan dilayani secara simultan oleh sejumlah pengguna. Pada sebagian besar kerja sistem *software*, beban maksimal menggambarkan gabungan beberapa tipe transaksi. Selanjutnya *durability test* dilakukan pada kondisi operasi fisik yang ekstrem seperti temperatur yang tinggi, kelembaban, mengendara dengan kecepatan tinggi, sebagai detail persyaratan spesifikasi durabilitas. Jadi, dibutuhkan untuk *real-time firmware* yang diintegrasikan ke dalam sistem seperti sistem senjata, kendaraan *transport* jarak jauh, *Internet Of Things* (IOT) dan keperluan meterologi. Isu ketahan pada *firmware* terdiri dari respon *firmware* terhadap efek cuara seperti temperatur panas atau dingin yang ekstrem, debu, kegagalan operasi ekstrem karena kegagalan listrik secara tiba-tiba, loncatan arus listrik dan putusnya komunikasi secara tiba-tiba.

4. *Training usuability test*

Ketika sejumlah besar pengguna terlibat dalam sistem operasi, *training usability requirement* ditambahkan dalam agenda pengujian. Lingkup dari training *usability test* ditentukan oleh sumber yang dibutuhkan untuk melatih pekerja baru untuk memperoleh level perkenalan dengan sistem yang ditentukan atau untuk mencapai tingkat produksi tertentu. Detail dari pengujian ini, sama halnya dengan yang lain, didasarkan pada karakteristik pekerja. Hasil dari pengujian ini harus menginspirasi rencana dari kursus pelatihan dan *follow-up* serta memperbaiki sistem operasi *software*.

5. *Operational usability test*

Fokus dari pengujian ini adalah produktifitas operator, yang aspeknya terhadap sistem yang mempengaruhi *performance* dicapai oleh operator sistem. *Operational usability test* dapat dijalankan secara manual.

Revision class partitioning adalah sebuah metode *black box* lainnya yang merupakan faktor dasar yang menentukan keberhasilan paket suatu *software*, pelayanan jangka panjang, dan keberhasilan penjualan ke sejumlah besar populasi pengguna. Berkaitan dengan hal tersebut terdapat tiga kelas pengujian revisi sebagai berikut:

1. *Reusability test*

Reusability menentukan bagian mana dari suatu program (modul, integrasi, dbs) yang akan dikembangkan untuk digunakan kembali pada proyek pengembangan *software* lainnya, baik yang telah direncanakan maupun yang belum. Bagian ini harus dikembangkan, disusun, dan didokumentasikan menurut prosedur perpustakaan *software* yang digunakan ulang.

2. *Software interoperability test*

Software interoperability berkaitan dengan kemamputan *software* dalam memenuhi perlengkapan dan paket *software* lainnya agar memungkinkan untuk mengoperasikannya bersama dalam satu sistem komputer kompleks.

3. *Equipment interoperability test*

Equipment interoperability berkaitan dengan perlengkapan *firmware* dalam menghadapi untuk perlengkapan lain dan atau paket *software*, dimana persyaratan mencantumkan *specified interfaces*, termasuk dengan *interfacing standard*. Pengujian yang relevan harus menguji implementasi dari *interoperability reqirements* dalam sistem.

Keuntungan dan kekurangan dari *black box testing*, beberapa keuntungan dari *black box testing*, diantaranya sebagai berikut:

1. *Black box testing* memungkinkan kita untuk memiliki sebagian besar tingkat pengujian, yang sebagian besarnya dapat diimplementasikan.
2. Untuk tingkat pengujian yang dapat dilakukan baik dengan *white box testing* maupun *black box testing*, *black box testing* memerlukan lebih sedikit sumber dibandingkan dengan yang dibutuhkan oleh *white box testing* pada pake *software* yang sama.

Sedangkan kekurangan dari *black box testing* adalah sebagai berikut:

1. Adanya kemungkinan untuk terjadinya beberapa kesalahan yang tidak disengaja secara bersama-sama akan menimbulkan respon pada pengujian ini dan mencegah deteksi kesalahan (*error*). Dengan kata lain, *black box test* tidak siap untuk mengidentifikasi kesalahan-kesalahan yang berlawanan satu sama lain sehingga menghasilkan *output* yang benar.

2. Tidak adanya kontrol terhadap *line coverage*. Pada kasus dimana *black box test* diharapkan dapat meningkatkan *line converage*, tidak ada cara yang mudah untuk menspesifikasikan parameter-parameter pengujian yang dibutuhkan untuk meningkatkan *coverage*. Akibatnya, *black box test* dapat melakukan bagian penting dari baris kode, yang tidak ditangani oleh set pengujian.

3. Ketidakmungkinan untuk menguji kualitas pembuatan kode dan pendekatannya dengan standar pembuatan kode.