BAB 2
TINJAUAN PUSTAKA

2.1 Ruang Lingkup Objek Penelitian

Objek penelitian yang diteliti yaitu SKK Migas (Satuan Kerja Khusus Pelaksana Kegiatan Usaha Hulu Minyak dan Gas) tepatnya di divisi fasilitas dan keuangan. SKK Migas beralamat di Wisma Mulia Lantai 35, Jl. Gatot Subroto Kav. 42, RT.03/RW.02, Kuningan Barat, Kota Jakarta Selatan, Daerah Khusus Ibukota Jakarta.

2.1.1 Sejarah Instansi

Satuan Kerja Khusus Pelaksana Kegiatan Usaha Hulu Minyak dan Gas Bumi (disingkat: SKK Migas) merupakan institusi yang dibentuk oleh pemerintah Republik Indonesia yang didirikan pada tanggal 10 Januari 2013 melalui peraturan presiden (Perpres) Nomor 9 Tahun 2013 tentang Penyelenggaraan Pengelolaan Kegiatan Usaha Hulu Minyak dan Gas Bumi. Badan ini menggantikan lembaga sebelumnya yaitu Badan Pelaksana Kegiatan Usaha Hulu Minyak dan Gas Bumi (disingkat: BPMIGAS). SKK Migas bertugas melaksanakan pengelolaan kegiatan usaha hulu minyak dan gas bumi berdasarkan Kontrak Kerja Sama. Pembentukan lembaga ini dimaksudkan supaya pengambilan sumber daya alam minyak dan gas bumi milik Negara dapat memberikan manfaat dan penerimaan yang maksimal bagi Negara untuk seebesar-besar kemakmuran rakyat.

2.1.2 Visi

Visi merupakan pandangan jauh tentang suatu perusahaan ataupun lembaga yang juga dapat diartikan sebagai tujuan perusahaan. Adapun visi dari SKK Migas yaitu:

Menjadi entitas yang proaktif dan terpercaya serta penggerak utama pengembangan industri strategis hulu minyak dan gas bumi bagi kepentingan bangsa dan Negara.

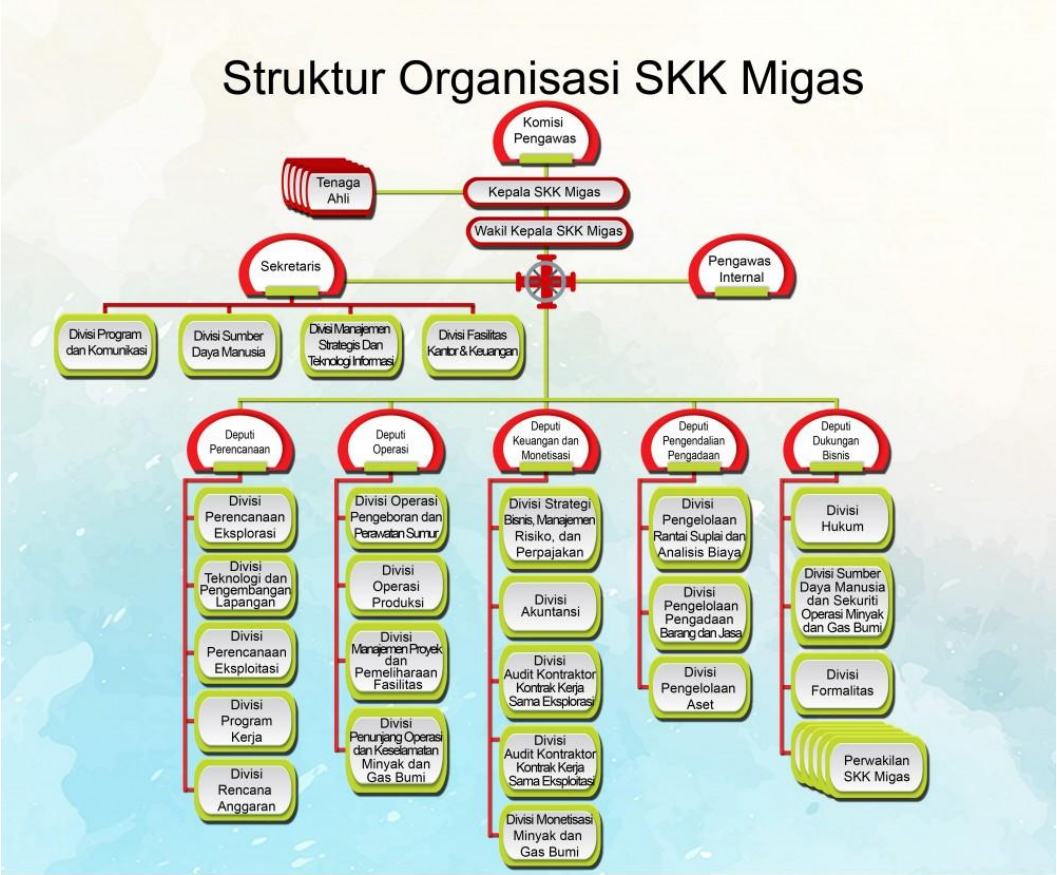
2.1.3 Misi

Untuk mencapai tujuan dan alasan mengapa SKK Migas dibuat maka terdapat misi yaitu:

1. Melakukan kegiatan pengawasan dan pengendalian kontrak kerja sama kegiatan usaha hulu minyak dan gas bumi (migas) untuk menjamin efektivitas, efisiensi, dan tetap menjaga kelestarian lingkungan hidup.
2. Melakukan sinergi dengan pemangku kepentingan dan Kontraktor Kontrak Kerja Sama (KKS) untuk meningkatkan cadangan dan produksi migas Indonesia.
3. Meningkatkan budaya kerja yang kondusif melalui sinergi, koordinasi, serta penerapan sistem manajemen perubahan, ilmu pengetahuan dan teknologi.
4. Mendukung dan menumbuhkembangkan kemampuan nasional untuk lebih mampu bersaing di tingkat nasional, regional, dan internasional.
5. Meningkatkan pendapatan Negara untuk memberikan kontribusi yang sebesar-besarnya bagi perekonomian nasional dan mengembangkan serta memperkuat posisi industri hulu migas Indonesia.

2.1.4 Struktur Organisasi

Struktur Organisasi SKK Migas dapat dilihat pada Gambar 2.1



Gambar 2.1 Struktur Organisasi SKK Migas

Penelitian dan pembangunan sistem evaluasi kendaraan dinas operasional ini dilakukan di Divisi Fasilitas Kantor dan Keuangan.

2.2 Landasan Teori

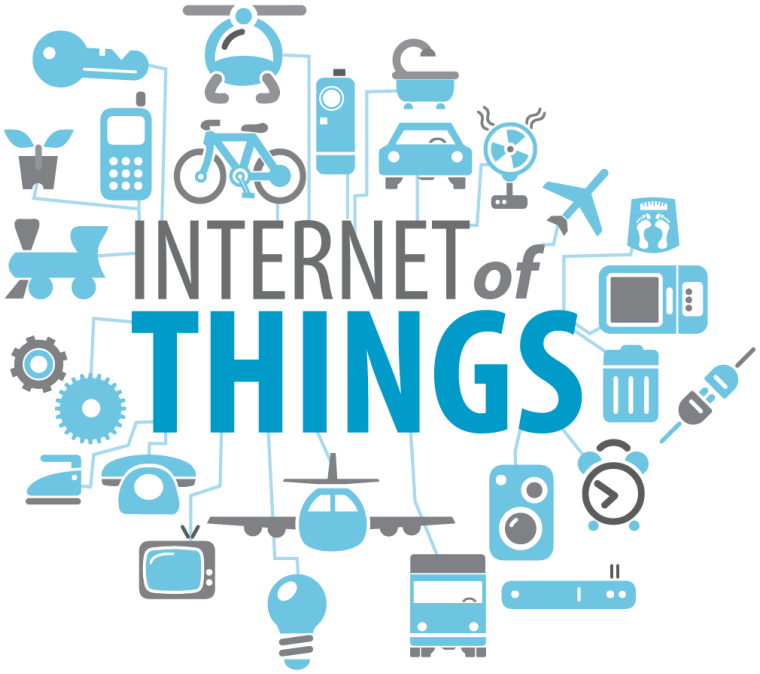
Landasan teori merupakan kumpulan dari beberapa definisi dan konsep yang disusun secara sistematis mengenai variable-variabel dalam penelitian. Landasan teori ini akan menjadi acuan dasar penulis dalam melakukan penelitian melalui uraian-uraian secara teoritis.

2.2.1 Internet of Things

Saat ini *Internet of Things* (IoT) mendapatkan banyak perhatian dari peneliti karena menjadi teknologi yang menjanjikan kehidupan manusia yang lebih pintar dengan konsep kemampuan berkomunikasi antar obyek, mesin dan apapun berdampingan dengan manusia. IoT merepresentasikan sebuah sistem

yang mana terdiri suatu benda di dunia nyata serta sensor-sensor yang terpasang atau dikombinasikan dengan benda tersebut, terkoneksi dengan Internet maupun itu melalui struktur jaringan kabel atau nirkabel.

Sensor-sensor pada IoT dapat menggunakan koneksi yang bermacam-macam seperti RFID, Wi-Fi, Bluetooth dan ZigBee. Selain itu untuk menjangkau area koneksi yang lebih jauh dapat menggunakan teknologi seperti GSM, GPRS, 3G dan LTE. Benda yang sudah terpasang IoT dapat bertukar informasi apapun yang didapat dari sensor. Dengan adanya teknologi IoT dunia akan menjadi *smart* dalam berbagai aspek melihat bahwa IoT menyediakan sarana yang mendukung *Smart City*, *Smart Healthcare*, *Smart Home and Building* dan banyak lagi penerapan penting seperti *Smart Energy* dan *Smart Transportation*[3].

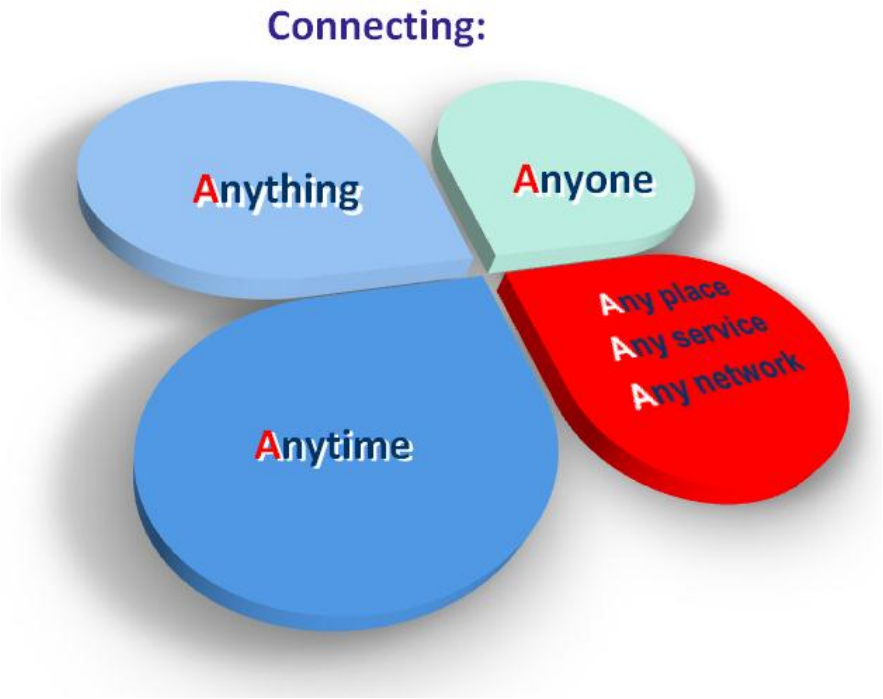


Gambar 2.2 *Internet of Things* (IoT)

2.2.1.1 Konsep Internet of Things

Dengan *Internet of Things* apapun dapat berkomunikasi dengan satu sama lain melalui jaringan kapanpun darimanapun untuk menyediakan layanan apapun dengan jaringan apapun untuk siapa saja. Konsep ini akan membangun sebuah tipe aplikasi yang baru yang kedepannya dapat tercapai *Smart Vehicle* dan *Smart*

Home untuk menyediakan layanan-layanan seperti pemberitahuan dini, keamanan, penghematan energy, otomasi, komunikasi, komputer dan hiburan[4].

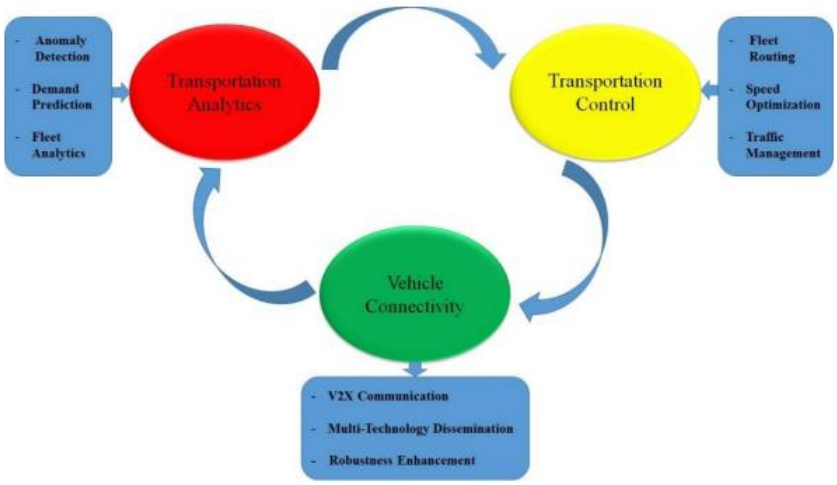


Gambar 2.3 Konsep *Internet of Things*

2.2.1.2 Internet of Things Untuk Transportasi

Pengembangan pada bidang transportasi adalah salah satu dari banyak faktor yang mengindikasikan kesejahteraan suatu Negara. Contohnya monitoring kondisi jalan dan sistem pemberitahuan dini adalah satu dari banyak aplikasi transportasi penting berbasis IoT[5]. Gambaran besar konsep dari *Smart Transportation* adalah untuk menerapkan prinsip urun daya dimana suatu informasi dikumpulkan dari banyak orang. Proses nya dimulai dari pengguna mengidentifikasi rute yang diinginkan dan ditampilkan beberapa tanda seperti jalan berlubang pada aplikasi *smartphone* [6]. *Smart Transportation* ideal dengan tiga konsep utama seperti digambarkan pada Gambar 2.4 yaitu diantaranya analisa transportasi, pengatur transportasi dan keterhubungan kendaraan. Analisa transportasi menguraikan analisis dari prediksi permintaan dan deteksi anomali. Perutean kendaraan dan pengaturan kecepatan sebagai tambahan untuk pengelolaan lalu lintas semua itu dikenal sebagai pengelolaan transportasi yang

mana sebenarnya sangat terkait bagaimana kendaraan-kendaraan terkoneksi (V2X *communication*).



Gambar 2.4 Aspek *Smart Transportation*

2.2.1.3 Tantangan Internet of Things

Pada kenyataannya bahwa penerapan-penerapan *Internet of Things* dan skenario-skenario yang dibahas diatas sangat menarik yang mana menyediakan suatu teknologi untuk membuat apapun menjadi *smart*. Tetapi, ada beberapa tantangan dalam implementasi *Internet of Things*. Dengan harapan bahwa teknologi harus tersedia secara murah dengan banyak perangkat yang terpasang, IoT juga menghadapi beberapa masalah lainnya [7], seperti:

1. *Scalability*

   Internet of Things mempunyai konsep lebih besar daripada komputer yang terhubung dan berkomunikasi melalui jaringan.
2. *Self-Organizing*

   Benda apapun yang diterapkan IoT seharusnya tidak lagi dikelola seperti komputer yang memerlukan pengguna untuk bekerja. Benda yang *smart* harus mampu untuk mengatur dan mengkonfigurasi dirinya sendiri untuk menyesuaikan dengan lingkungan dimana ditempatkan.
3. *Software Complexity*

Infrastruktur perangkat lunak akan sangat dibutuhkan dengan tujuan mengelola perangkat-perangkat pintar dan menyediakan layanan untuk mendukungnya. Hal ini dikarenakan sistem perangkat lunak pada perangkat yang *smart* haruslah berfungsi dengan sumber daya yang minimal.

4. *Security and Privacy*

   Internet of Things mempunyai berbagai resiko keamanan bagi consumer maupun bisnis. Pelaku kejahatan dunia maya bisa saja mengakses perangkat atau menangkap jaringan komunikasi untuk mengambil informasi penting. Mereka bisa saja menyerang server-server atau server berbasis *cloud* yang mana data yang terkumpul sangat banyak menjadi target yang menarik bagi mereka. Salah satu mengurangi resiko yaitu dengan melakukan enkripsi tetapi terkendala oleh rendahnya daya dan kapasitas memproses pada perangkat sensor. Dan juga untuk meminimalisir adanya pencurian dan penyalahgunaan data oleh pihak yang bertanggung jawab maka sangat disarankan untuk hanya mengumpulkan data yang diperlukan saja untuk tujuan yang jelas.

5. *Fault Tolerance*

   Perangkat pada *Internet of Things* lebih dinamis dan portabel daripada komputer dan dapat berubah secara drastis dengan cara yang tidak terduga. Hal ini sangat mungkin untuk terjadinya kesalahan. Maka dibutuhkan suatu sistem yang memastikan perangkat dapat kembali beroperasi walaupun mengalami kegagalan.

6. *Power Supply*

   Perangkat pada *Internet of Things* kadangkala bergerak dan tidak terknoneksi dengan listrik, jadi pengoperasiannya memerlukan tenaga dari sumber yang cukup. Penghematan energy merupakan faktor tidak hanya pada perangkat keras dan arsitektur sistem, tetapi juga pada perangkat lunak.

7. Komunikasi Nirkabel (*Wireless Communication)*

Dari sudut pandang penggunaan energi, teknologi nirkabel seperti GSM, UMTS, Wi-Fi dan Bluetooh kurang cocok digunakan karena konsumsi daya. Teknologi terbaru seperti ZigBee dan lainnya masih dikembangkan yang memungkinkan komunikasi nirkabel tetapi dengan lebih hemat daya.

2.2.1.4 IoT Untuk Diagnostik Kendaraan

Dengan memonitor semua aspek pada kendaraan maka akan lebih mudah untuk mendeteksi masalah yang ada dengan mengirimkan semua informasi pada sensor-sensor yang tertanam ke server pusat tersertifikasi dimana teknisi-teknisi dapat mengaplikasikan keahliannya untuk menemukan dan memprediksi kesalahan pada kendaraan [8].

Gambar 2.5 Konsep *Connected-Car*

Disamping bagian diagnostik, sistem ini dapat membantu pengemudi untuk terus memantau pemantauan, mengatur penetapan servis atau menganalisa konsumsi bahan bakar dan memberikan anjuran untuk gaya mengendarai yang lebih *eco-friendly*.

Pada situasi yang sulit seperti kecelakaan mengendara, sistem dapat mampu untuk memberitahu layanan darurat pertolongan ketika pengendara tidak mampu, meningkatkan kemungkinan bagi yang terluka untuk hidup dengan mengurangi waktu keseluruhan misi penyelamatan. Untuk sektor bisnis atau

pemerintahan dapat memberikan cara yang ditingkatkan untuk memantau seluruh kendaraan perusahaan menggunakan protokol terbuka dan mengurangi biaya yang berasal dari kebiasaan yang sudah ada solusinya di pasar.

2.2.2 Basis Data

Basis Data adalah kumpulan informasi yang disimpan dalam komputer secara sistematik utuk memperoleh informasi dari basis data. Basis data adalah representasi kumpulan fakta yang saling berhubungan disimpan secara sistematik, bersama untuk memenuhi berbagai kebutuhan.

Basis data atau *database* merupakan sekumpulan informasi yang saling berkaitan pada suatu subjek tertentu untuk tujuan tertentu pula. Basis data atau *database* adalah susunan *record* data operasional lengkap dari suatu organisasi atau perusahaan, yang diorganisasi dan disimpan secara terintegrasi dengan menggunakan metode tertentu sehingga mampu memenuhi informasi yang optimal yang dibutuhkan oleh para penggunanya.

Istilah basis data atau "*database*" berawal dari ilmu komputer, kemudian artinya semakin luas, memasukkan hal-hal yang di luar bidang elektronika. Catatan yang mirip dengan basis data atau *database* sebenarnya sudah ada sebelum revolusi industri, yaitudalam bentuk buku besar, kuitansi dan kumpulan data yang berhubungan dengan bisnis [9].

2.2.2.1 Perangkat untuk Membuat Basis Data

Basis data dapat dibuat dan diolah dengan menggunakan suatu program komputer, yaitu *software* (perangkat lunak). *Software* yang digunakan untuk mengelola dan memanggil *query* pada basis data disebut *database management system* (DBMS) atau sistem manajemen basis data.

DBMS terdiri atas dua komponen, yaitu *Relational Database Management System* (RDBMS) dan *Overview of Database Management System* (ODBMS). RDBMS meliputi *interface drivers, SQL Engine*, *transaction engine* dan *storage engine*. Adapun *Overview of Database Management System* (ODBMS) meliputi *language drivers, query engine*, *transaction engine* dan *storage engine*.

Adapun tingkatan dari perangkat lunak nya, terdapat dua tingkatan perangkat lunak yang memungkinkan untuk membuat sebuah basis data, antara lain: *high-level software* dan *low-level software*. *High level Software* meliputi Microsoft SQL Server, Oracle, Sybace, Interbase, Xbase, Firebird, MySQL, PostgreSQL, Microsoft Accees, Dbase III, Parados, Foxpro, Visual Foxpro, Arago Force, Recital, Dbfast, Dbxl, Quicksilver, Clipper, Flagship, Harbour, Visual Dbase, dan Lotus Smart Suite Approach. Adapun yang termasuk dalam *Low Level Software* antara lain Btrieve dan Tsunami Record Manager.

2.2.2.2 Karakteristik Basis Data

Karakteristik basis data atau *database* dalam DBMS memiliki tiga karakteristik utama, yaitu:

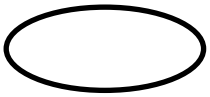
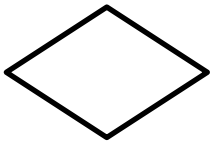
1. Data yang sama dapat diakses secara serempak oleh beberapa pengguna untuk berbagai kegunaan yang berbeda.
2. Data tidak bergantung pada struktur penyimpanan atau cara membaca data dari program palikasi, atau data bersifat transparan terhadap program aplikasi.
3. Data memiliki integritas (akurasi dan validasi) yang terkendali. Strategi akses terhadap data yang besifat logis menyebabkan basis data atau *database* berbeda dari *file-file* komputer yang lain. Aplikasi sangat bergantung pada struktur data yang dimiliki oleh *database*. Oleh karena itu, aplikasi tidak mengetahui cara data disimpan pada basis data. Basis data memiliki sifat terbebaskan dari keadaan fisik data yang dapat menyebabkan basis data atau *database* mengembang ukurannya, tetapi hal tersebut dapat mudah diatasi dengan cara dipindahkan ke sistem yang lebih besar lgai tanpa menulis ulang aplikasinya.

2.2.2.3 Entity Relationship Diagram (ERD)

ERD atau *Entity Relationship Diagram* merupakan suatu model data yang dibentuk berdasarkan objek. ERD digunakan untuk menjelaskan hubungan antar data dalam sebuah basis data kepada pengguna secara logis. ERD didasarkan pada suatu persepsi bahwa dunia nyata terdiri atas objek-objek dasar tersebut.

Penggunaan ERD relatif mudah dipahami, bahkan oleh para pengguna yang awam. Bagi perancang atau analis sistem, ERD berguna untuk memodelkan sistem yang nantinya akan mengembangkan basis data. Model ini juga membantu perancang atau analis sistem pada saat melakukan analis dan perancangan basis data karena model ini dapat menunjukkan macam data yang dibutuhkan dan hubungan antar data didalamnya [10]. Komponen ERD atau *Entity Relationship Diagram* adalah sebagai berikut:

Tabel 2.1 Komponen Penyusun ERD

Komponen	Keterangan
	Persegi panjang mewakili entitas
	Elips mewakili atribut
	Belah ketupat mewakili relasi
	Garis menghubungkan relasi dan entitas atau relasi dan entitas dengan atributf

2.2.3 Business Process Model and Notation (BPMN)

Business Process Model and Notation (BPMN) adalah sebuah standar untuk memodelkan proses bisnis yang menyediakan notasi grafis dalam menjelaskan sebuah proses bisnis di dalam sebuah *Business Process Diagram* (BPD). Teknik aliran pada BPMN sama persis dengan *Activity Diagram* Pada UML. Tujuan dari BPMN adalah untuk mendukung manajemen proses bisnis, baik untuk pengguna teknis dan pengguna bisnis, dengan menyediakan notasi yang intuitif bagi pengguna bisnis, namun mampu mewakili proses semantik yang kompleks. Tujuan yang paling utama dari BPMN adalah untuk menyediakan sebuah standar notasi yang mudah di mengerti oleh semua pelaku bisnis. Termasuk para analisis bisnis yang membuat dan menyempurnakan proses bisnis, pengembang yang bertanggung jawab mengimplementasikan proses bisnis

tersebut dan manajer bisnis yang memantau dan mengelola proses bisnis. Sehingga BPMN mengatasi perbedaan pemahaman yang terjadi antara perancang dan pelaksana dalam sebuah proses bisnis.

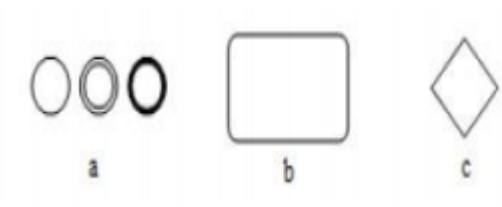
2.2.3.1 Elemen-Elemen BPMN

BPMN mendefinisikan sebuah *Business Process Diagram* (BPD), menggunakan dasar dari teknik flowchart yang disesuaikan untuk membuat model grafis dari operasi proses bisnis. Sebuah model proses bisnis adalah sebuah jaringan dari objek grafis yang terdiri dari aktivitas-aktivitas dan aturan alur yang mendefinisikan urutan kejadian[11].

Terdapat 4 Kategori dalam elemen *Business Process Diagram* pada BPMN, yaitu:

1. *Flow Object* (Objek aliran)
2. *Connecting Objects* (Objek Penghubung)
3. *Swimlanes*
4. *Artifacts*

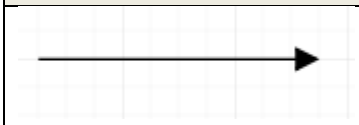
Dalam *Business Process Diagram* (BPD) terdapat 3 elemen inti pada *Flow Objects* yang dapat dilihat pada Gambar 2.6 yaitu *Event, Activity* dan juga *Gateway*. Hal ini memudahkan pengguna untuk memodelkan dikarenakan tidak perlu mempelajari dan mengenali banyak elemn yang berbeda. Sebauh *Event* disimbolkan dengan sebuah lingkaran yang menunjukan terjadinya suatu proses. Suatu *Event* dapat mempengaruhi aliran proses dan biasanya mempunyai penyebab (*trigger*) atau sebuah dampak (*result*). Ada 3 jenis *Event* yaitu *Start, Intermediate* dan *End*. Sebuah *Activity* disimbolkan dengan sebuah persegi dengan sudut yang tidak memiliki radius. Pada *Activity* dapat berupa *Task* yang bersifat tunggal atau kata kerja ataupun *Sub-process* yang dapat dibagi lagi menjadi beberapa *Activity*. Sebuah *Sub-Process* disimbolkan dengan tanda tambah di bagian tengah bawah persegi. *Gateway* merupakan elemen yang disimbolkan dengan bentuk intan dan digunakan untuk mengatur pemisahan atau penyatuan dari beberapa aliran urutan. Tanda di dalam sebuah *Gateway* dapat mengindikasikan jenis dari sifat *Gateway* tersebut.

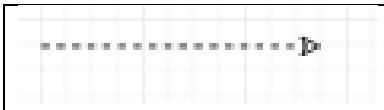
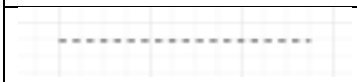


Gambar 2.6 Elemen-elemen BPMN

Objek aliran terhubung satu sama lain didalam sebuah diagram untuk membuat struktur kerangka dasar dari proses bisnis seperti pada Gambar 2.6, (a) *Event*, (b) *Activity/Task*, (c) *Gateway*. Ada 4 jenis objek yang dapat menghubungkan untuk fungsi ini yaitu *Sequence Flow*, *Message Flow*, *Data Association* dan *Association*. *Sequence Flow* digunakan untuk menunjukan urutan aktifitas yang terjadi dalam sebuah proses. *Message Flow* digunakan untuk menunjukkan aliran pesan antara 2 partisipan proses yang terpisah, sedangkan *Association* digunakan untuk menghubungkan data, teks atau artifak lain dengan objek aliran.

Tabel 2.2 Komponen Penghubung BPMN

Penghubung	Nama	Fungsi
	*Sequence Flow*	Digunakan untuk menggambarkan urutan dari elemen-elemen didalam proses dan model koreografi
	*Message Flow*	Digunakan untuk menampilkan aliran pesan antara dua partisipan yang dapat mengirim dan menerimanya.

	*Data Association*	Digunakan untuk menggambarkan aliran informasi antar *activity* dalam suatu proses bisnis
	Association	Digunakan untuk menghubungkan artifak-artifak dengan elemen-elemen BPMN (gambar).

2.2.4 Unified Modeling Language (UML)

UML *(Unified Modeling Language*) merupakan sebuah bahasa yang telah menjadi standar dalam industri untuk visualisasi, merancang dan mendokumentasikan sistem perangkat lunak. UML mempunyai sebuah standar untuk merancang model sebuah sistem. Dengan memanfaatkan UML kita dapat membuat model untuk semua jenis perangkat lunak, dimana apikasi tersebut dapat berjalan pada perangkat keras, sistem operasi dan jaringan apapun, serta ditulis dalam bahasa pemrograman apapun. UML lebih cocok digunakan untuk pengembangan perangkat lunak yang menggunakan object oriented, seperti: C++, Java, C#, atau VB.NET, karena UML merupakan bahasa yang menggunakan class dan operation dalam konsep dasarnya. Walaupun demikian, UML tetapi dapat digunakan untuk memodelkan aplikasi procedural dalam VB atau C.

UML mendefinisikan notasi dan syntax/semantic. Notasi UML merupakan sekumpulan bentuk khusus untuk meggambarkan berbagai diagram perangkat lunak. Setiap bentuk memiliki makna tertentu, dan UML syntax mendefinisikan bagaimana bentuk-bentuk tersebut dapat dikombinasikan. Notasi UML terutama diturunkan dari tiga notasi yang telah ada sebelumnya: Grady Booch OOD (Object-Oriented Design), Jim Rumbaugh OMT (Object Modeling Technique), dan Ivar Jacobson (Object-Oriented Software Engineering)[12].

2.2.4.1 Diagram UML

UML menyediakan 10 (sepuluh) macam diagram untuk memodelkan aplikasi berorientasi object, yaitu:

1. *Use Case Diagram* berfungsi untuk memodelkan proses bisnis
2. *Conceptual Diagram* berfungsi untuk memodelkan konsep-konsep yang ada di dalam aplikasi
3. *Sequence Diagram* untuk memodelkan pengiriman pesan (*message*) antar objek
4. *Collaboration Diagram* untuk memodelkan interaksi antar objek
5. *State Diagram* untuk memodelkan perilaku objek di dalam sistem
6. *Activity Diagram* untuk memodelkan perilaku user dan objek di dalam sistem
7. *Class Diagram* untuk memodelkan struktur kelas
8. *Object Diagram* untuk memodelkan struktur objek
9. *Component Diagram* untuk memodelkan komponen objek
10. *Deployment Diagram* untuk memodelkan distribusi aplikasi

Dari 10 (sepuluh) macam diagram yang disediakan UML, terdapat 3 (tiga) diagram yang paling sering digunakan dalam pembangunan aplikasi berorientasi objek, yaitu *use case diagram*, *activity diagram* dan *class diagram*.

2.2.4.2 Use Case Diagram

Use Case Diagram adalah titik awal yang sangat baik untuk hampir setiap aspek pembangunan sistem berorientasi objek, desain, pengujian, dan dokumentasi. *Use Case Diagram* menggambarkan sistem persyaratan ketat dari luar ke dalam, dan juga menentukan nilai yang sistem memberikan kepada pengguna.

2.2.4.3 Activity Diagram

Activity Diagram sangat baik jika digunakan pada pemodelan proses bisnis. *Activity Diagram* sangat membantu untuk mengkoordinasi kita hingga mencapai tujuan bisnis.

2.2.4.4 Class Diagram

Class Diagram merupakan diagram yang selalu ada di permodelan berorientasi objek, karena class diagram menunjukan hubungan antar class dalam sistem yang sedang dibangun dan bagaimana mereka saling berkolaborasi untuk mencapai suatu tujuan.

2.2.5 Web Service

Web Service adalah salah satu bentuk sistem perangkat lunak yang didesain untuk mendukung interaksi mesin ke mesin melalui jaringan. Sistem *web service* memungkinkan sebuah fungsi di dalam *web service* dapat diakses oleh aplikasi lain tanpa perlu mengetahui detail pemrograman yang terdapat didalamnya[13].

Secara umum arsitektur *web service* terdiri dari 3 komponen, yaitu:

1. *Service Provider*, merupakan pemilik dari *web service* yang berfungsi menyediakan kumpulan operasi dari *web service.*
2. *Service Requestor*, merupakan aplikasi yang bertindak sebagai klien dari *web service* yang mencari dan memulai interaksi terhadap layanan yang disediakan.
3. *Service Registry*, merupakan tempat dimana *Service Provider* mempublikasikan layanannya. Pada arsitektur *Web Service*, *service registry* bersifat opsional. Teknologi *web service* memungkinkan kita dapat menghubungkan berbagai jenis perangkat lunak yang memiliki platform dan sistem operasi yang berbeda.

Ada 2 jenis *web service* yaitu REST dan SOAP. SOAP (*Simple Object Access Protocol*) merupakan protokol untuk saling bertukar pesan antar aplikasi. Spesifikasi format pesan tersebut didefinisikan seperti amplop berbentuk XMLyang dikriim beserta aturan-aturan atau cara untuk menerjemahkan representasi data dari XML.

Cara kerja SOAP ialah aplikasi klien mengirim request berbentuk XML kepada provider *web service*. *Web Service* kemudian menerima permintaan

(*request*) tersebut, menjalankan layanan (*service*) kemudian mengirimkan balasan (*response*) ke aplikasi klien juga dalam bentuk XML. Baik permintaan (*request*) mapupun balasan(*response*) keduannya menggunakan protokol SOAP.

REST adalah singkatan dari *Representational State Transfer* merupakan *web service* yang menerapkan konsep perpindahan antar *state* dimana dalam bernavigasi REST menggunakan *link* HTTP untuk melakukan aktifitas tertentu. Dalam pengaplikasiannya REST banyak digunakan untuk *web service* yang berorientasi pada *resource*. Maksud orientasi pada *resource* adalah orientasi yang menyediakan *resource* sebagai layanannya dan bukan kumpulan dari aktifitas yang mengolah *resource* itu. *Response* dari *web service* REST dapat berupa XML atau JSON (*Javascript Object Notation*).

2.2.6 Arduino

Arduino adalah mikrokontroler *open-source* yang dapat dengan mudah diprogram, dihapus dan diprogram ulang dengan waktu yang singkat. Diperkenalkan pada tahun 2005 platform Arduino dirancang untuk menyediakan cara yang murah untuk yang hobi, pelajar sampai professional untuk membangun perangkat yang dapat berinteraksi dengan lingkungan menggunakan sensor dan aktuator [14].

Gambar 2.7 Arduino Zero

Arduino mampu untuk terhubung dengan jaringan Internet dan bertukar informasi dengan perangkat lain dengan menggunakan modul tambahan seperti modul GPS/GSM.

2.2.7 GPS Receiver

Global Position System (GPS) adalah sebuah sistem navigasi yang dapat memberikan informasi mengenai posisi suatu alat yang mampu menerima sinyal pemosisian dengan memanfaatkan teknologi satelit. Alat yang dapat digunakan untuk menerima sinyal tersebut dinamakan *GPS Receiver*. Dengan memanfaatkan modul GPS dapat diperoleh posisi suatu objek atau perangkat yang sudah ditanamkan alat *GPS Receiver*[15].

Gambar 2.8 GPS *Receiver* – GP-20U7

Informasi yang diperoleh dari perangkat *GPS Receiver* yaitu, posisi lintang yang dinamakan *latitude*, dan posisi bujur atau *longitude*. Berdasarkan posisi lintang dan posisi bujur yang didapatkan *GPS Receiver* data dapat diolah ke dalam bentuk yang lebih berguna, seperti contohnya dengan menggunakan map digital yang dapat menggambarkan secara jelas posisi akurat sehingga posisi dari suatu objek atau perangkat dapat dengan mudah diketahui.

Banyak perangkat yang dapat digunakan sebagai GPS Receiver, salah satunya adalah Arduino. Untuk menjadikan Arduino sebagai GPS Receiver, perlu disiapkan sebuah modul GPS. Contoh modul GPS yaitu Skylab SKM53 dengan antenna internal. Pada modul GPS terdapat enam pin, tetapi biasanya hanya empat pin yang digunakan, yaitu:

1. VCC (dihubungkan dengan pin 5V di Arduino)
2. GND (dihubugnkan dengan pin GND di Arduino)
3. TXD (dihubungkan dengan salah satu pin digital di Arduino)
4. RXD (dihubungkan dengan salah satu pin digital di Arduino)

2.2.8 GSM

GSM adalah singkatan dari *Global System For Mobile Communication*, merupakan salah satu protokol telepon seluler yang menjadi standar di sebagian besar belahan dunia. Protokol GSM diciptakan pada tahun 1980-an dan 90-an untuk membakukan layanan telepon seluler antara Negara Negara Eropa. Perangkat yang memanfaatkan GSM menggunakan *subscriber identify module* (SIM) card, yang mana penting untuk fungsi mereka dan memungkinkan pengguna mengganti ponsel dengan mudah[16].

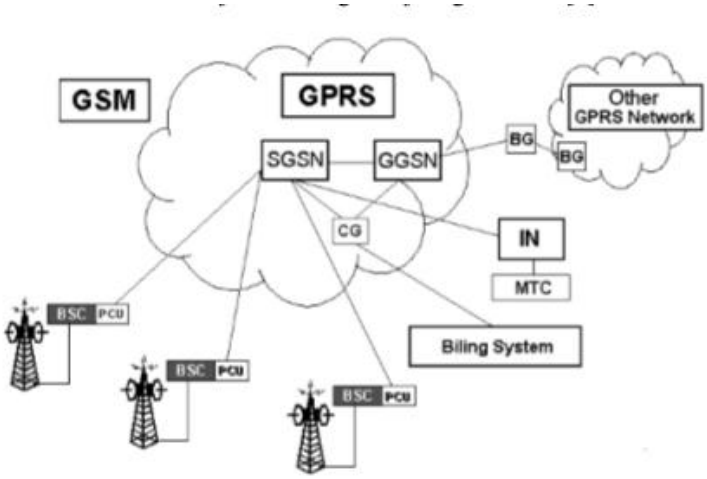
2.2.8.1 Frekuensi GSM

Sistem GSM adalah sistem frekuensi dengan *time-division* di tiap masing-masing fisiknya. Chanel yang ditandai dengan frekuensi pembawa dan sejumlah slot waktu. Frekuensi GSM umunnya meiputi dual band di 900 MHz dan 1800 MHz, atau umumnya disebut sebagai GSM-900.

2.2.9 GPRS

General Packet Radio Service atau disingkat GPRS merupakan sistem transmisi berbasis paket untuk GSM (*Global System For Mobile Communication*) yang menggunakan prinsip "*Tunneling*" dan menawarkan kecepatan transfer data yang lebih tinggi dengan kisaran mencapai 160Kbps. Pada segi pembiayaan, teknologi GPRS mengacu pada besar *volume* yang digunakan baik mengirim maupun menerima[17].

Berikut dibawah ini menunjukan konfigurasi jaringan GPRS:



Gambar 2.9 Konfigurasi jaringan GPRS

2.2.10 Akselerometer

Sensor *accelerometer* adalah sebuah tranduser yang digunakan untuk mengukur percepatan linear, mengukur dan mendeteksi getaran, dan mengukur percepatan akibat gravitasi. Sensor *accelerometer* mengukur percepatan akibat gerakan benda yang melekat padanya [18].

Gambar 2.10 Modul Akselerometer

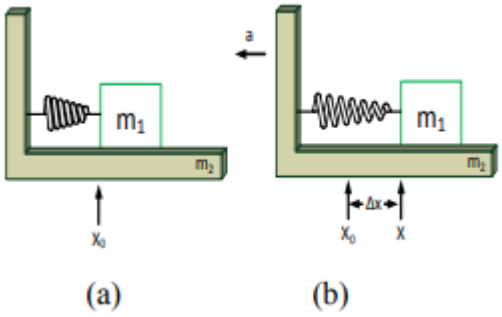
Prinsip kerja dari tranduser ini berdasarkan hokum fisika bahwa apabila suatu konduktor digerakkan melalui suatu medan magnet, atau jika suatu medan

magnet digerakkan melalui suatu konduktor, maka akan timbul suatu tegangan induksi pada konduktor tersebut. *Accelerometer* yang diletakan di permukaan bumi dapat mendeteksi percepatan 1g (ukuran gravitasi bumi) pada titik vertikalnya, untuk percepatan yang dikarenakan oleh pergerakan horizontal maka *accelerometer* akan mengukur percepatannya secara langsung ketika bergerak secara horizontal.

Akselerometer adalah sensor yang digunakan untuk mengukur percepatan atu perubahan kecepatan setiap waktu. Sensor ini dipasang bersama benda yang akan diukur akselerasinya, seperti mengukur perubahan kecepatan roket yang meluncur atau digunakan untuk analisis getaran (vibration analysis) pada mesin, serta digunakan utuk mendeteksi gerak dan kemiringan pada *smartphone*.

2.2.10.1 Konsep Akselerometer

Akselerometer dapat dianalogikan sebagai sebuah sistem massa-pegas (*mass spring system*) ynag bekerja berdasarkan Hukum Newton dan Hukum Hooke. Prinsip kerja dari sensor ini akan dijelaskan sebagai berikut.



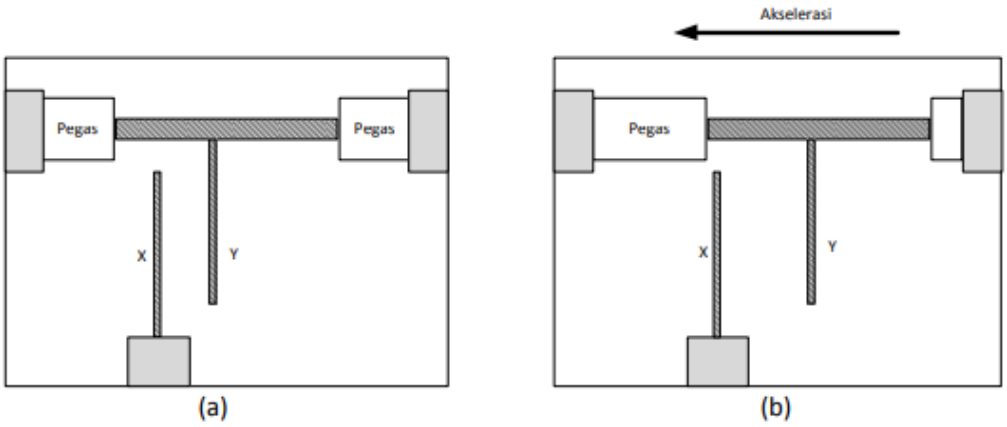
Gambar 2.11 Sistem masa pegas sebagai akselerometer.
a. Sistem dalam keadaan normal, b. Pegas meregang karena akselerasi

2.2.10.2 Sensor Akselerometer Elektronik dengan Teknologi MEMS

Sensor akselerometer elektronik adalah sensor akselerometer yang hasil pengukuran akselerasionya dinyatakan dalam tegangan atau data digital. Seperti penjelasan sebelumnya bahwa akselerometer dapat dibangun dengan masa yang dikaitkan dengan pegas, akselerometer elektronik memiliki prinsip yang sama dalam mengukur percepatan hanya saja tidak mungkin untuk membuat sensor

dengan ukuran yang relative besar seperti pada Gambar 2.11. Hingga pada akhir nya pada akhir abad ke 20 dikembangkan teknologi MEMS atau juga disebut dengan *Micro-Electro-Mechanical Systems*, yang mampu menerapkan prinsip akselerometer massa-pegas ke dalam sebuah *chip* yang kecil.

Akselerometer dengan teknologi MEMS memanfaatkan perubahan kapasitansi dua buah play terhadap perubahan jarak antar plat tersebut karena pengaruh akselerasi dari luar. Prinsip kerja dari akselerometer kapasitif ini dijelaskan pada Gambar 2.12 dibawah ini.



Gambar 2.12 Struktur Akselerometer Elektronik

a. Posisi normal tanpa percepatan
b. Jarak X-Y merengang karena akselerasi

Pada Gambar 2.3, terdapat plat yang tetap (Y) dan plat yang dapat bergerak secara elastis (X). saat sistem mendapatkan akselerasi (Gambar 2.3b), jarak antara kedua plat ini akan berubah dan menyebabkan kapasitansi kedua plat juga berubah. Selanjutnya dengan rangkaian elektronik perubahan kapasitansi ini diubah menjadi tegangan yang proporsional dengan akselerasi eksternal yang dirasakan oleh sistem.

2.2.11 PostgreSQL

PostgreSQL adalah sistem basis data objek-relasional sumber terbuka yang kuat yang menggunakan dan memperluas bahasa SQL digabung dengan banyak fitur yang secara aman menyimpan dan menskalasikan beban kerja paling rumit.

Asal muasal PostgreSQL adalah waktu mundur ke tahun 1986 dimana sebagian dari proyek POSTGRES di University of California di Berkeley dan memiliki lebih dari 30 tahun pengembangan aktif di platform inti [19].

Gambar 2.13 Logo PostgreSQL

PostgreSQL telah mendapatkan reputasi yang kuat untuk arsitekturnya yang sudah terbukti, handal, integritas data, fitur-fitur yang kuat, kemampuan untuk diperbesar serta dedikasi komunitas sumber terbuka di balik layar perangkat lunak untuk secara konsisten memberikan solusi yang performant dan inovatif. PostgreSQL berjalan di semua jenis sistem operasi utama, telah sesuai dengan ACID sejak tahun 2001 dan memiliki *add-ons* yang kuat seperti PostGIS yang populer dalam mengelola basis data geospasial.

PostgreSQL hadir dengan banyak fitur yang bertujuan untuk membantu pengembang membangun aplikasi, administrator untuk melindungi integritas data dan membangun lingkungan yang toleran terhadap kesalahan serta membantu untuk mengelola data terlepas dari seberapa besar skala data yang akan disimpan. Selain gratis dan bersumber terbuka, PostgreSQL sangat dapat dikembangkan. Misalnya, menentukan tipe data sendiri, membuat fungsi khusus, bahkan menulis kode dari bahasa pemrogramman yang berbeda tanpa mengkompilasi ulang database.

PostgreSQL mencoba untuk menyesuaikan dengan standar SQL dimana kesesuaian tersebut tidak bertentangan dengan fitur tradisonal atau dapat

menyebabkan keputusan arsitektur yang buruk. Banyak fitur yang diperlukan oleh standar SQL yang didukung meskipun kadang-kadang dengan sintaks atau fungsi yang sedikit berbeda. Langkah lebih lanjut menuju kesesuaian dapat diharapkan dari waktu ke waktu. Pada rilis versi 10 pada bulan Oktober 2017 PostgreSQL memenuhi setidaknya 160 dari 179 fitur wajib untuk SQL dimana pada tulisan ini belum ada basis data relasional yang memenuhi kesesuaian dengan standar ini.

Berikut adalah berbagai fitur PostgreSQL dengan lebih banyak lagi yang ditambahkan setiap rilis mayor.

1. Tipe Data
    a. Primitives: Integer, Numeric, String, Boolean
    b. Structured: Date/Time, Array, Range, UUID
    c. Document, JSON/JSONB, XML, Key-value (Hstore)
    d. Geometry, Point, Line, Circle, Polygon
    e. Customizations: Composite, Custom Type
2. Integritas Data
    a. UNIQUE, NOT NULL
    b. Primary Keys
    c. Foreign Keys
    d. Exclusion Constraints
    e. Explicit Locks, Advisory Locks
3. Konkurensi, Performa
    a. Indexing: B-tree, multicolumn, Expressions, Partial
    b. Advanced Indexing: GIST, SP-Gist, KNN Gist, GIN, BRIN, Bloom Filters
    c. Sophisticated query planner/optimizer
    d. Multi-Version concurrency control (MVCC)
    e. Parallelization of read queries
    f. Table Partitioning
4. Kehandalan, Pemulihan Bencana
    a. Write-ahead logging (WAL)

b. Replication: Asynchronous, Synchronous, Logical
c. Point-in-time-recovery (PITR), active standbys
d. Tablespaces

5. Keamanan
   a. Authentication: GSSAPI, SSPI, LDAP, SCRAH-SHA-256, Certificate, dan lainnya.
   b. Sistem Akses kontrol yang kuat
   c. Keamanan level kolom atau baris
6. Kemungkinan Dikembangkan
   a. Stored Procedures
   b. Procedural Language: PL/PGSQL, Perl, Python dan lainnya.
   c. Koneksi ke database lain dengan antarmuka SQL standar
   d. Banyak ekstensi yang dapat menyediakan fitur lain seperti PostGIS
7. Internasionalisasi, Pencarian Teks
   a. Mendukung set karakter internasional
   b. Pencarian teks-penuh

2.2.11.1 PostGIS

PostGIS adalah ekstensi basis data berelasi objek PostgreSQL yang memungkinkan obyek GIS (*Geographic Information System*) untuk disimpan pada basis data. PostGIS dikembangkan oleh perusahaan Refractions Research Inc, sebagai proeyk teknologi basis data spasial. Perusahaan Refractions adalah perusahaan konsultan GIS dan basis data di Victoria, British Columbia, Kanada yang dikhususkan pada integrasi data dan pengembangan perangkat lunak [20].

Gambar 2.14 Logo PostGIS

2.2.12 NodeJS

NodeJS adalah sebuah *runtime* JavaScript yang dibuat dalam mesin javascript Chrome's V8. Node.js menggunakan model *event-driven* dan *non-blocking* I/O yang membuatnya ringan dan efisien. Node.js mempunyai ekosistem *package* yang bernama Npm (Node Package Manager) yang merupakan ekosistem terbesar dari pustaka sumber terbuka didunia [21].

Sebagai *runtime* JavaScript yang mendukung *Asynchronous*, Node.js dirancang untuk membangun aplikasi jaringan yang skalabel. Berikut adalah contoh program sederhana dimana koneksi ditangani secara konkuren:

```
const http = require('http');

const hostname = '127.0.0.1';
const port = 3000;

const server = http.createServer((req, res) => {
  res.statusCode = 200;
  res.setHeader('Content-Type', 'text/plain');
  res.end('Hello World\n');
});

server.listen(port, hostname, () => {
  console.log(`Server aktif http://${hostname}:${port}/`);
});
```

Hal ini berbeda dengan model konkurensi yang lebih umum saat ini dimana *thread* pada OS digunakan. Jaringan berbasis *thread* relatif kurang efisien dan sangat sulit digunakan. Selain itu, pengguna Node.js bebas dari kekhawatiran dari proses yang *dead-lock* mengingat Node.js tidak ada *lock*. Hampir tidak ada fungsi didalam Node.js secara langsung melakukan operasi I/O, jadi proses tidak akan pernah terhenti. Karena tidak blok maka pengembangan sistem sangat masuk akal untuk dikembangan menggunakan Node.js.

2.2.13 ReactJS

ReactJS adalah *library* javascript sumber-terbuka yang bertanggung jawab hanya untuk mengatur tampilan dari aplikasi. ReactJS dibuat dan dimaintenance oleh Facebook. ReactJS menggunakan mekanisme *Virtual DOM* untuk memperbaharui tampilan pada website. *Virtual DOM* bekerja secara cepat karena hanya mengubah elemen DOM (*Document Object Model*) individu daripada

melakukan *reload* keseluruhan struktur DOM (*Document Object Model*) setiap kali.

ReactJS memungkinkan kita untuk membuat komponen menggunakan bahasa domain khusus yang disebut JSX. JSX memperbolehkan kita untuk menulis komponen menggunakan HTML dengan menggabungkannya dengan Javascript. ReactJS akan secara internal mengubahnya kedalam *virtual dom* dan akan menghasilkan output berupa HTML [22].

2.2.13.1 Instalasi dan Pengaturan

ReactJS adalah pustaka Javascript yang terkandung dalam satu buah file react-<versi>.js yang dapat disertakan didalam halaman HTML mana saja. Biasanya react-dom-<version>.js juga disertakan bersamaan dengan pustaka utama ReactJS.

1. Instalasi Dasar

```
<!DOCTYPE html>
<html>
 <head></head>
 <body>
 <script type="text/javascript" src="/path/ke/react.js"></script>
 <script type="text/javascript" src="/path/ke/react-dom.js"></script>
 <script type="text/javascript">
 // Gunakan kode react JavaScript disni atau didalam file terpisah
 </script>
 </body>
</html>
```

Pustaka JavaScript dapat didapatkan dari halaman instalasi pada dokumentasi resim React .

React juga mendukung sintaks JSX. JSX adalah sebuah ekstensi yang dibuat oleh Facebook yang menambahkan sintaks XML pada JavaScript. Untuk menggunakan JSX diperlukan menyertakan pustaka Babel dan mengubah: `<script type=”text/javascript">`

Menjadi:

`<script type=”text/babel”/>`

untuk menerjemahkan sintaks JSX ke kode JavaScript.

```
<!DOCTYPE html>
<html>
 <head></head>
 <body>
 <script type="text/javascript" src="/path/ke/react.js"></script>
 <script type="text/javascript" src="/path/ke/react-dom.js"></script>
 <script src="https://npmcdn.com/babel-core@5.8.38/browser.min.js"></script>
 <script type="text/babel">
 // Gunakan kode JSX react disini atau didalam file terpisah
 </script>
 </body>
</html>
```

2. Instalasi via NPM

ReactJS juga dapat diinstall menggunakan npm (Node Package Manager) dengan cara berikut

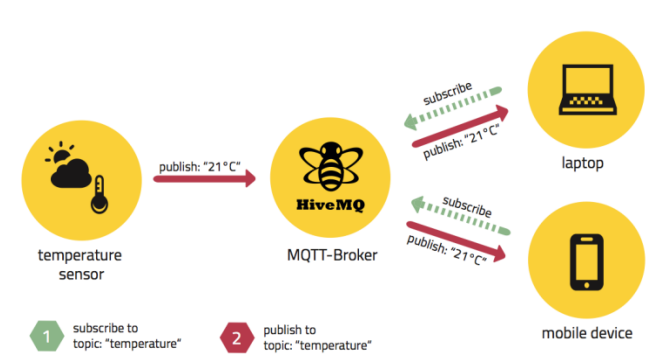
⇨ npm install –save react react-dom

Untuk menggunakan ReactJS pada proyek JavaScript dapat dilakukan sebagai berikut:

```
var React = require('react');
var ReactDOM = require('react-dom');
ReactDOM.render(<App />, ...);
```

2.2.14 MQTT (Message Queueing Telemetry Transport)

MQTT atau *Message Queuing Telemetry Transport* adalah protokol pesan *publish/subscribe* yang sangat sederhana dan ringan. Arsitektur MQTT dirancangan terbuka dan mudah untuk diimplementasi dengan sampai seribu klien dapat ditangani oleh hanya satu *server*. Karakteristik ini membuat MQTT ideal untuk digunakan dalam lingkup kerja yang terbatas dimana *bandwith* jaringannya rendah atau terdapat latensi yang tinggi dan dengan perangkat yang mungkin memiliki kemampuan memproses dan memorinya yang terbatas[23].



Gambar 2.15 Konsep Protokol MQTT

Beberapa manfaat dari protokol MQTT yaitu:

1. Memperluas konektifitas antar batas perusahaan dengan perangkat pintar.
2. Menawarkan konektifitas yang optimal untuk sensor-sensor dan perangkat *remote*.
3. Memberikan data yang relevan untuk aset pengambilan keputusan yang pintar.
4. Memungkinkan skalabilitas penyebaran yang luas dan manajemen solusi.

MQTT meminimalisir *bandwith* jaringan dan sumber daya perangkat yang dibutuhkan selagi memastikan reabilitas dan pengiriman. Pendekatan ini membuat protokol MQTT cocok untuk mengkoneksikan antar mesin dengan mesin (M2M) yang mana menjadi aspek kritikal dalam konsep Internet of Things (IoT) saat ini.

Protokol MQTT memiliki keunggulan lain yaitu:

1. Terbuka dan bebas rolayti untuk adopsi yang mudah
   MQTT terbuka untuk membuatnya mudah diadopsi untuk macam-macam perangkat.
2. Model *push/subscribe messaging* yang memfasilitasi distribsi satu-ke-banyak.
   Mengirimkan aplikasi atau perangkat tidak perlu mengetahui apapun akan penerima, atau bahkan alamatnya.
3. Ideal untuk jaringan yang terbatas (*low bandwith*, latensi tinggi, keterbatasan data dan koneksi yang kurang baik.

Message header yang digunakan MQTT dibuat seminimal mungkin. *Header* hanya berukuran dua bytes dan sesuai permintaan, distribusi pesan dengan gaya *push* menjaga penggunaan jaringan serendah mungkin.

4. Didesain khusus untuk perangkat *remote* dengan daya pemrosesan dan memori yang rendah.

   Header yang minimal dan sedikitnya ketergantungan pada *library* membuat MQTT ideal untuk perangkat yang terbatas.

5. Mudah untuk digunakan dan diimplementasikan dengan perintah-perintah pesan yang sederhana.

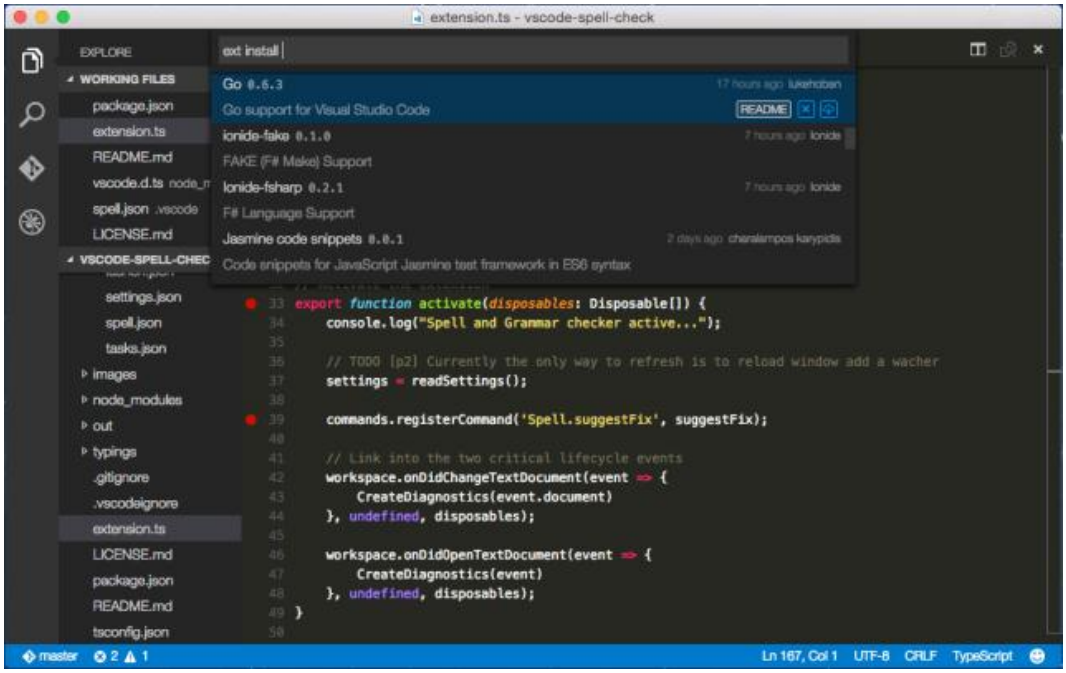
   Banyak aplikasi dari MQTT yang dapat dicapai dengan hanya perintah CONNECT, PUBLISH, SUBSCRIBE dan DISCONNECT.

6. Dukungan yang sudah ada akan kehilangan kontak antara client dengan server.

   Server akan diberitahu ketika sebuah klien tidak bekerja, memungkinkan pesan untuk dikirimkan ulang atau disimpan untuk pengiriman nanti.

2.2.15 Visual Studio Code

Visual Studio Code adalah editor kode yang dikembangkan oleh Microsoft untuk Windows, Linux dan MacOS. Visual Studio Code mendukung *debugging*, control Git, *syntax highlighting*, *intelligent code completion*, *snippets*, dan *code refactoring*. Visual Studio Code gratis dan *open-source* [24].



Gambar 2.16 Visual Studio Code

2.2.16 Arduino IDE

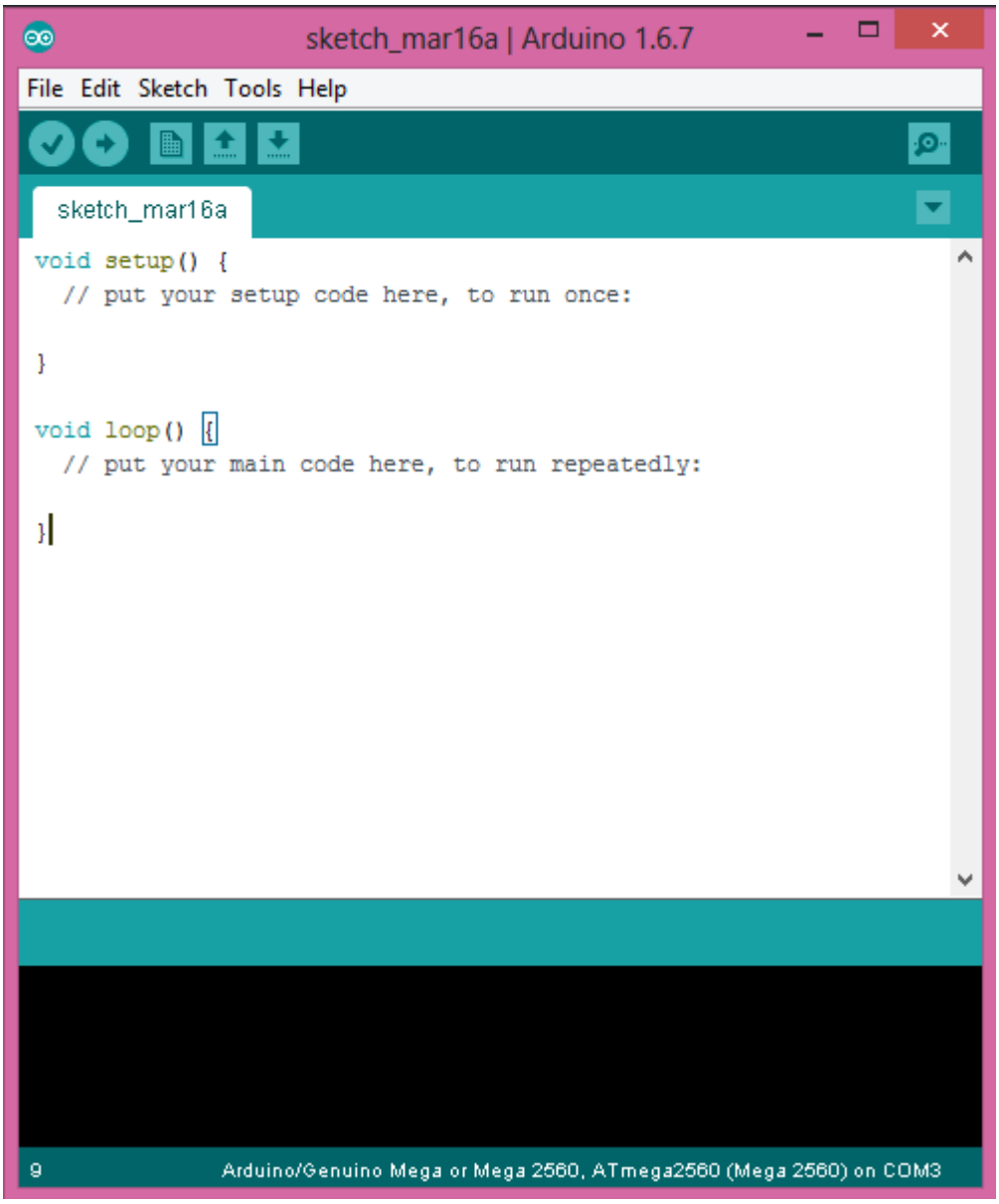
IDE merupakan kependekan dari Intergrated Development Environment, atau merupakan suatu lingkungan terintegrasi yang digunakan untuk melakukan pengembangan. Disebut sebagai lingkungan karena melalui software inilah Arduino dilakukan pemrograman untuk melakukan fungsi-fungsi yang dibenamkan melalui sintaks pemrograman. Arduino menggunakan bahasa pemrogramman sendiri yang menyerupai bahasa C. Arduino IDE dibuat dari bahasa pemrograman JAVA. Arduino IDE juga dilengkapi dengan *library* C/C++ yang biasa disebut dengan *Wiring* yang membuat operasi input dan output menjadi lebih mmudah. Arduino IDE ini dikembangkan dari software Processing yang dirombak menjadi Arduino IDE khusus untuk pemrograman dengan Arduino [25].

2.2.16.1 Menulis Program

Program yang ditulis dengan menggunakan Arduino Software (IDE) disebut dengan Sketch. Sketch ditulis dalam suatu editor teks dengan disimpan dalam file dengan ekstensi ".ino". Teks editor pada Arduino IDE memiliki fitur-

fitur seperti *cutting*/*paste* dan *searching*/*replacing* sehingga memudahkan dalam menulis kode program.

Pada aplikasi Arduino IDE, terdapat semacam message box berwarna hitam yang berfungsi menampilkan status seperti pesan *error*, *compile*, dan *upload* program. Di bagian bawah paling kanan Arduino IDE menunjukan board yang terkonfigurasi beserta COM Ports yang digunakan.



Gambar 2.17 Tampilan Editor Arduino IDE

Berikut tombol-tombol toolbar yang tersedia pada editor Arduino IDE

Verify

Berfungsi untuk melakukan *checking* kode yang telah ditulis apakah sudah sesuai dengan kaidah pemrograman yang ada atu belum

Upload

Berfungsi untuk melakukan kompilasi program atau kode yang sudah ditulis menjadi bahasa yang dapat dipahami oleh mesin dan menuliskanya pada papan Arduino.

New

Berfungsi untuk membuat Sketch baru.

Open

Berfungsi untuk membuka *sketch* yang pernah dibuat dan membuka kembali untuk dilakukan perubahan atau sekedar *upload* ulang ke Arduino

Save

Berfungsi untuk menyimpan Sketch yang telah dibuat.

Serial Monitor

Berfungsi untuk membuka serial monitor. Serial monitor disini merupakan jendela yang dapat menampilkan data apa saja yang dikirimkan atau dipertukarkan antara arduino dengan Sketch pada port serialnya. Serial Monitor ini sangat berguna sekali ketika membuat program atau melakukan *debugging* tanpa menggunakan LCD pada Arduino. Serial monitor ini dapat digunakan untuk menampilkan suatu nilai proses, nilai pembacaan, bahkan pesan *error*.

Terdapat menu pada Editor yaitu:

1. File
   a. New, berfungsi untuk membuat sketch baru dengan *bare minimum* yang terdiri dari void setup() dan void loop()
   b. Open, berfungsi membuka sketch yang pernah dibuat di dalam *drive*

c. Open Recent, merupakan menu yang berfungsi untuk mempersingkat waktu membuat file atau Sketch yang terakhir dibuka.
d. Skeetchbook, berfungsi menunjukan hirarki Sketch yang dibuat termasuk struktur foldernya
e. Example, berisi contoh-contoh pemrograman yang disediakan pengembang Arduino, sehingga dapat mempelajari program-program dari contoh yang diberikan.
f. Close, berfungsi untuk menutup jendela Arduino IDE dan menghentikan aplikasi
g. Save, berfungsi untuk menyimpan Sketch yang dibuat atau perubahan yang dilakukan pada Sketch
h. Save as..., berfungsi menyimpan Sketch yang sedang dikerjakan atau Sketch yang sudah disimpan dengan nama yang berbeda
i. Page Setup, berfungsi mengatur tampilan pada proses pencetakan
j. Print, berfungsi untuk mengirimkan file Sketch ke mesin cetak untuk dicetak
k. Preferences, Berfungsi untuk merubah pengaturan Arduino IDE
l. Quit, berfungsi menutup semua jendela Arduino IDE. Sketch yang masih terbuka pada saat tombol Quit ditekan secara otomatis akan terbuka lagi pada saat Arduino IDE dijalankan.

2. Edit
   a. Undo/Redo, berfungsi untuk mengembalikan perubahan yang sudah dilakukan pada Sketch beberapa langkah mundur dengan Undo atau maju dengan Redo.
   b. Cut, befungsi untuk menghapus teks yang terpilih pada editor dan menempatkan teks tersebut pada *clipboard*.
   c. Copy, berfungsi menduplikasi teks yang terpilih kedalam editor dan menempatkan teks tersebut pada *clipboard*.
   d. Copy for Forum, berfungsi melakukan copy kode dari editor dan melakukan *formatting* agar sesuai untuk ditampilkan pada forum,

sehingga kode tersebut bisa digunakan sebagai bahan diskusi dalam forum.

e. Paste, berfungsi menyalin data yang terdapat pada *clipboard*, kedalam editor
f. Select All, berfungsi untuk melakukan pemilihan teks atau kdoe dalam halaman editor
g. Comment/Uncomment, berfungsi memberikan atau menghilangkan tanda "//" pada kode atau teks, dimana tanda tersebut menjadi suatu baris kode sebagai komen dan tidak disertakan pada tahap kompilasi.
h. Increase/Decrease Indent, berfungsi untuk mengurangi atau menambahkan indentasi pada baris kode tertentu. Indentasi adalah "Tab"
i. Find, berfungsi memanggil jendela window find and replace, dimana dapat digunakan untuk menemukan teks atau kata tertentu dalam program atau menemukan serta menggantikan kata tersebut dengan kata lain
j. Find Next, berfungsi menemukan kata setelahnya dari kata pertama yang berhasil ditemukan
k. Find Previous, befungsi menemukan kata sebelumnya dari kata pertama yang berhasil ditemukan

3. Sketch
   a. Verify/Compile, berfungsi untuk mengecek apakah Sketch yang kamu buat ada kekeliruan dari segi sintaks atau tidak. Jika tidak ada kesalahan maka sintaks yang dibuat akan dilakukan *compile* kedalam bahasa mesin.
   b. Upload, berfungsi mengirimkan program yang sudah dikompilasi ke Arduino Board.
   c. Upload Using Programmer, menu ini berfungsi untuk menuliskan *bootloader*.

d. Export Compiled Binary, berfungsi untuk menyimpan file dengan ekstensi .hex, dimana file ini dapat disimpan sebagai arsip untuk di upload ke board lain menggunakan tools yang berbeda
e. Show sketch Folder, berfungsi membuka folder Sketch yang sedang dikerjakan
f. Include Library, berfungsi menambahkan library/pustaka kedalam Skketch yang dibuat dengan menyertakan sintaks #include di awal kode. Selain itu dapat juga menambahkan library eksternal dari file .zip kedalam Arduino IDE.
g. Add File.., berfungsi untuk menambahkan file kedalam Sketch Arduino (file akan di-*copy* dari drive asal). File akan muncul sebagai tab baru dalam jendela Sketch.

4. Tools
   a. Auto Format, berfungsi melakukan pengaturan format kode pada jendela editor
   b. Archive Sketch, berfungsi menyimpan sketch kedalam file .zip
   c. Fix Encoding & Reload, berfungsi memperbaiki kemungkinan perbedaan antara pengkodean peta karakter editor dan peta karakter sistem operasi yang lain
   d. Serial Monitor, berfungsi membuka jendela serial monitor untuk melihat pertukaran data.
   e. Port, memilih port sebagai kanal komunikasi antara software dengan hardware
   f. Board, berfungsi memilih dan melakukan konfigurasi board yang digunakan
   g. Programmer, menu ini digunakan ketika hendak melakukan pemrogramman chip mikrokontroler tanpa menggunakan koneksi Onboard USB-Serial. Biasanya pada proses burning *bootloader*.
   h. Burn Bootloader, memungkinkan untuk menulis program bootloader kedalam IC mikrokontroler.

5. Help, berfungsi untuk mendapatkan bantuan terhadap masalah yang timbul ketika menggunakan Arduino IDE. Menu help ini terdapat file-file dokumentasi yang berkaitan dengan masalah yang sering muncul serta penyelesaianya. Selain itu pada menu help juga diberikan link untuk menjuju Arduino Forum guna menanyakan serta mendiskusikan berbagai masalah yang ditemukan.

2.2.17 Pengujian Black-Box

Pengujian *Black-Box*, juga disebut pengujian *behavioral* yang berfokus pada kebutuhan atau persyaratan fungsional dari perangkat lunak. Artinya, teknik pengujian *Black-Box* memungkinkan untuk mendapatkan set kondisi masukan yang sepenuhnya akan melaksanakan semua persyaratan fungsional untuk suatu program [26].

Pengujian *Black-Box* bukan merupakan alternatif dari pengujian *White-Box.* Melinkan, pengujian *Black-Box* adalah pendekatan komplementer yang mungkin untuk mengungkap kelas yang berbeda dari kesalahan daripada metode pengujian *White-Box.*

Pengujian *Black-Box* mencoba untuk menemukan kesalahan dalam kategori berikut:

1. Fungsi tidak benar atau hilang.
2. Kesalahan *interface* atau antarmuka
3. Kesalahan dalam struktur data attau akses basis-data eksternal
4. Kesalahan kinerja atau perilaku
5. Kesalahan inisialisasi dan terminasi

Tidak seperti pengujian *White-Box* yang dilakukan pada awal proses pengujian, pengujian *Black-Box* cenderung diterapkan pada tahap selanjutnya dari pengujian. Karena pengujian *Black-Box* sengaja mengabaikan struktur kontrol, perhatian difokuskan pada domain informasi. Pengujian ini dirancang untuk menjawab pertanyaan berikut

1. Bagaimana validitas fungsional diuji ?

2. Bagaimana perilaku sistem dan kinerja diuji ?
3. Apakah kelas masukkan akan membuat kasus uji yang baik ?
4. Apakah sistem sangat sensitif terhadap nilai masukkan tertentu ?
5. Bagaimana batas-batas kelas data yang terisolasi ?
6. Kecepatan data dan volume data apa yang dapat mentolerir sistem ?
7. Efek apa yang akan muncul dari kombinasi data tertentu terhadap operasi sistem ?

Salah satu dari pengujian *Black-Box* yang dapat dapat dilakukan oleh seorang penguji independen adalah pengujian fungsional. Basis uji dari pengujian fungsional ini adalah pada spesifikasi dari komponen perangkat lunak yang akan diuji. Pengujian fungsional memastikan bahwa seua kebutuhan-kebutuhan telah dipenuhi dalam sistem aplikasi. Dengan demikian fugnsinya adalah tugas-tugas yang didesain untuk dilaksanakan sistem. Pengujian fungsional berkonsentrasi pada hasil dari proses, bukan bagaimana prosesnya terjadi.