BAB 2
TINJAUAN PUSTAKA

2.1 Profil instansi

Bike.Bdg telah di kelola oleh komunitas BCCF (Bandung Creative City Forum) selama kurang lebih 3 tahun terhitung dari pertama di luncurkan pada tanggal 10 juni 2012 saat itu walikota Bandung Ridwan Kamil mengemukakan harapannya "Program bike sharing harus dapat menjadi solusi dari kemacetan kota Bandung dan juga sebagai alternatif pendukung peningkatan aspek pariwisata". Kini Bike.bdg mempunyai infrastruktur, 10 shelter penyewaan sepeda , 20 petugas pencatat penyewaan sepeda dan 150 sepeda yang bisa disewakan kepada masyarakat.

2.1.1 Sejarah instansi

Bike sharing adalah sebuah sistem layanan transportasi umum berupa sepeda yang dapat disewakan untuk perseorangan sebagai transportasi sehari-hari. Menyediakan sepeda yang dapat digunakan sebagai transportasi alternatif selain kendaraan bermotor yang diharapkan dapat berdampak baik terhadap pengurangan polusi udara dan suara serta kepadatan lalu-lintas. Dalam penyelenggaraan program ini, komunitas Bike.Bdg didukung penuh oleh BCCF (Bandung Creative City Forum) dengan Ridwan Kamil, ketua BCCF, sebagai penasihat program. Selain itu, program bike sharing jg didukung oleh Ikatan Alumni ITB. Untuk melakukan penyewaan harus memiliki kartu identitas, seperti Kartu Tanda Penduduk(KTP) atau Kartu Tanda Mahasiswa(KTM), datang ke shelter Bike.bdg terdekat, berikan KTP atau KTM untuk diproses oleh shelter, pilih sepeda yang anda inginkan dan sesuai kebutuhan anda. Ambil kembali kartu identitas yang telah di proses. Bayar tariff penyewaan setelah melakukan proses penyewaan di lakukan. Tarif penyewaan sepeda untuk hari biasa Rp. 3000/jam, sedangkan untuk car freeday Rp. 10.000/jam. Shelter Bike.Bdg mulai beroperasi dari pukul 05.00 WIB hingga pukul 18.00 WIB.

2.1.2 Visi dan Misi

Berikut adalah visi dan misi dari Bike.Bdg :

2.1.2.1 Visi

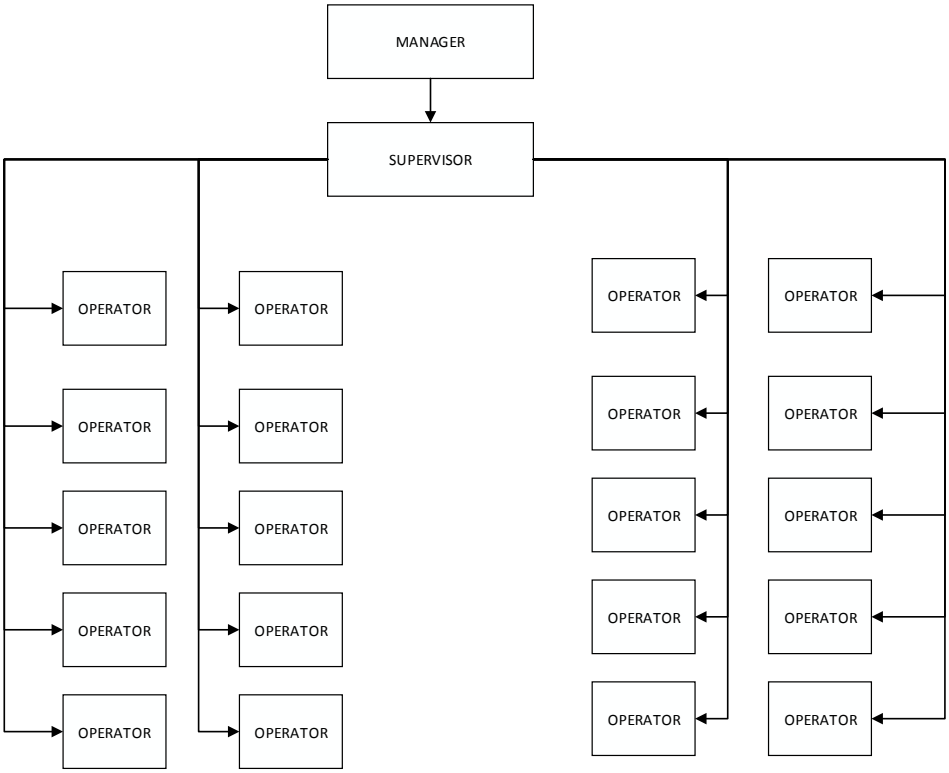
Visi dari Bike.Bdg yaitu menyediakan sepeda yang dapat digunakan sebagai transportasi alternatif selain kendaraan bermotor yang di harapkan dapat berdampak baik terhadap pengurangan polusi udara dan suara serta kepadatan lalu-lintas.

2.1.2.2 Misi

Berdasarkan visi yang telah dipaparkan diatas maka Misi Bike.Bdg adalah: misi dari Bike.Bdg yang cukup sederhana memberikan dampak besar, yaitu, menyediakan sepeda yang dapan di gunakan sebagai transportasi alternatif selain kendaraan bermotor.

2.1.3 Struktur Organisasi

Gambar 2.1 adalah struktur organisasi Bike.Bdg



Gambar 2.1 Struktur Organisasi Bike.Bdg

Tugas dan Wewenang

Tugas dan Wewenang dari struktur organisasi Bike.Bdg adalah sebagai berikut:

1. Manager

   Tugas dan tanggung jawab Manager adalah:

   a. Melakukan penyusunan rencana dan program kerja rental.
   b. Mengawasi kegiatan sistem kerja rental .
   c. Menerima laporan rental dan laporan keanggotaan dari supervisor

2. Supervisor

   Tugas dan tanggung jawab bagian Supervisor adalah:

   a. Memantau kinerja masing-masing operator.
   b. Menerima laporan dari operator dan menyerahkan laporan kepada manager

3. Operator

   a. Melayani konsumen yang akan melakukan peminjaman.
   b. Memberikan perintah pada karyawan bagian laporan.
   c. Memberikan laporan penyewaan dan laporan anggota kepada supervisor.

2.2 Landasan Teori

Landasan teori dari penulisan skripsi ini menguraikan proses analisis sistem serta mendukung proses pembangunan sistem monitoring penyewaan seepeda menggunakan teknologi RFID (Radio Frequency Identification) :

2.2.1 Sistem

Mendefinisikan sistem terdapat dua kelompok pendekatan sistem yang lebih menekankan pada prosedur dan elemennya. Prosedur didefinisikan sebagai suatu urutan yang tepat dari tahapan-tahapan intruksi yang menerangkan apa yang harus di kerjakan, siapa yang mengerjakan, kapan di kerjakan dan bagaimana mengerjakannya. Berdasarkan pendekatan elemen adalah sistem sebagai bagian-bagian yang saling berkaitan yang beroperasi bersama untuk mencapai beberapa sasaran atau maksud[4].

2.2.2 Informasi

Informasi adalah data yang telah diolah menjadi bentuk yang lebih Berarti dan berguna bagi penerimanya untuk mengambil keputusan masa kini maupun masa yang akan datang. Fungsi dari informasi adalah untuk mengurangi ketidakpastian didalam proses pengambilan keputusan tentang suatu keadaan[4].

2.2.3 Kualitas Sistem

Kualitas sistem merupakan karakteristik dari informasi yang Melekat mengenai sistem itu sendiri yang mana kualitas sistem merujuk Pada seberapa baik kemampuan perangkat keras, perangkat lunak, dan Kebijakan prosedur dari sistem informasi yang dapat menyediakan Informasi kebutuhan pemakai[5].

2.2.4 Kualitas Informasi

Kualitas informasi merupakan tingkat dimana informasi memiliki Karakteristik isi, bentuk, dan waktu, yang memberikannya nilai buat para Pemakai akhir tertentu untuk mendukung pengambilan keputusan dalam Suatu perusahaan/organisasi.

2.2.5 Skala likert

Skala likert adalah sebuah tipe skala psikometri yang menunjukan Persetujuan responden terhadap pertanyaan positif dianggap sama derajatnya dengan persetujuan responden terhadap pernyataan negatif dan Menggunakan angket atau menggunakan skala yang lebih luas dalam penelitian survei[6].

2.2.6 Monitoring dan Evaluasi

Monev adalah kegiatan *monitoring* dan evaluasi yang ditujukan pada suatu program yang sedang atau sudah berlangsung. *Monitoring* atau pemantauan adalah suatu seni mengumpulkan informasi dengan usaha yang minimum untuk membuat suatu keputusan di waktu yang tepat. Informasi yang sudah dikumpulkan dapat digunakan untuk kemudian dilakukan analisis, diskusi, maupun pembuatan laporan. *Monitoring* memungkinkan kita untuk memantau proses secara rutin dalam jangka waktu tertentu, menentukan apakah suatu proses tersebut sudah berjalan dengan baik atau sudah sesuai dengan tujuan awal.

Evaluasi adalah proses untuk mengidentifikasi masalah, mengumpulkan data dan menganalisis data, menyimpulkan hasil yang telah dicapai, menginterpretasikan hasil menjadi rumusan kebijakan, dan menyajikan informasi (rekomendasi) untuk pembuatan keputusan berdasarkan pada aspek kebenaran hasil evaluasi. Perbedaan antara *monitoring* dan evaluasi adalah monitoring dilakukan pada saat program masih berjalan sedangkan evaluasi dapat dilakukan baik sewaktu program itu masih berjalan ataupun program itu sudah selesai. Atau dapat juga bila dilihat dari pelakunya, monitoring biasanya dilakukan oleh pihak internal sedangkan evaluasi dilakukan oleh pihak internal maupun eksternal. Evaluasi dilaksanakan untuk memperoleh fakta atau kebenaran dari suatu program beserta dampaknya, sedangkan monitoring hanya melihat keterlaksanaan program, faktor pendukung dan faktor penghambatnya. [13]

2.2.7 Basis Data

Sistem basis data adalah sistem terkomputerisasi yang tujuan utamanya adalah memelihara data yang sudah diolah atau informasi dan membuat informasi tersedia saat dibutuhkan [7]. Pada intinya basis data adalah media untuk menyimpan data agar dapat diakses dengan mudah dan cepat.

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

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

2.2.7.1 DBMS

DBMS (*Database Management System*) atau dalam bahasa Indonesia sering disebut sebagai Sistem Manajemen Basis Data adalah suatu sistem aplikasi yang digunakan untuk menyimpan, mengelola, dan menampilkan data [7]. Suatu sistem aplikasi disebut DBMS jika memenuhi persyaratan minimal sebagai berikut :

1. Menyediakan fasilitas untuk mengelola akses data
2. Mampu menangani integritas data
3. Mampu menangani akses data yang dilakukan secara bersamaan
4. Mampu menangani *backup* data

2.2.7.2 SQL

SQL (*Structured Query Language*) adalah bahasa yang digunakan untuk mengelola data pada RDBMS[8]. SQL awalnya dikembangkan berdasarkan teori aljabar relasional dan kalkulus.

Meskipun SQL diadopsi dan diacu sebagai bahasa standar oleh hampir sebagian besar RDBMS yang beredar saat ini, tetapi tidak semua standar yang tercantum dalam SQL diimplementasikan oleh seluruh DBMS tersebut. Sehingga kadang-kadang ada perbedaan perilaku (hasil yang ditampilkan) oleh DBMS yang berbeda padahal *query* yang dimasukkan sama.

2.2.8 Pemrograman Berorientasi Objek

Metodologi berorientasi objek adalah suatu strategi pembangunan perangkat lunak yang mengorganisasikan perangkat lunak sebagai kumpulan objek yang berisi data dan operasi yang diberlakukan terhadapnya [8]. Metodologi berorientasi objek merupakan suatu cara bagaimana sistem perangkat lunak dibangun melalui pendekatan objek secara sistematis. Metode berorientas objek didasarkan pada penerapan prinsip-prinsip pengelolaan kompleksitas. Metode berorientasi objek meliputi rangkaian aktivitas analisis berorientasi objek,

perancangan berorientasi objek, pemrograman berorientasi objek, dan pengujian berorientasi objek.

Pada saat ini, metode berorientasi objek banyak dipilih karena metodologi lama banyak menimbulkan masalah seperti adanya kesulitan pada saat mentransformasi hasil dari satu tahap pengembangan ke tahap berikutnya, misalnya pada metode pendekatan terstruktur, jenis aplikasi yang dikembangkan saat ini berbeda dengan masa lalu. Aplikasi yang dikembangkan pada saat ini beragam dengan *platform* yang berbeda-beda, sehingga menimbulkan tuntutan kebutuhan metodologi pengembangan yang dapat mengakomodasi ke semua jenis aplikasi tersebut.

Keuntungan menggunakan metodologi berorientasi objek adalah sebagai berikut

1. Meningkatkan produktivitas

   Karena kelas dan objek yang ditemukan dalam suatu masalah masih dapat dipakai ulang untuk masalah lainnya yang melibatkan objek tersebut (*reusable*).

2. Kecepatan pengembangan

   Karena sistem yang dibangun dengan baik dan benar pada saat analisis dan perancangan akan menyebabkan berkurangnya kesalahan pada saat pengodean.

3. Kemudahan pemeliharaan

   Karena dengan model objek, pola-pola yang cenderung tetap dan stabil dapat dipisahkan dan pola-pola yang mungkin sering berubah-ubah.

4. Adanya konsistensi

   Karena sifat pewarisan dan penggunaan notasi yang sama pada saat analisis, perancangan maupun pengkodean.

5. Meningkatkan kualitas perangkat lunak

   Karena pendekatan pengembangan lebih dekat dengan dunai nyata dan adanya konsistensi pada saat pengembangannya, perangkat lunak yang dihasilkan akan mampu memenuhi kebutuhan pemakai serta mempunyai sedikit kesalahan.

2.2.8.1 Konsep Dasar Berorientasi Objek

Pendekatan berorientasi objek merupakan suatu teknik atau cara pendekatan dalam melihat permasalahan dan sistem (sistem perangkat lunak, sistem informasi, atau sistem lainnya) [8]. Pendekatan berorientasi objek akan memandang sistem yang akan dikembangkan sebagai suatu kumpulan objek yang berkorespondensi dengan objek-objek dunia nyata.

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

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

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

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

1. Kelas (*Class*)

   Kelas adalah sekumpulan objek-objek dengan karakteristik yang sama. Kelas merupakan definisi statis dan himpunan objek yang sama yang mungkin lahir atau diciptakan dan kelas tersebut. Sebuah kelas akan mempunyai sifat

(atribut), kelakuan (metode/operasi), hubungan (*relationship*) dan arti. Suatu kelas dapat diturunkan dan kelas yang lain, dimana atribut dan kelas semula dapat diwariskan ke kelas yang baru. Secara tekniks kelas adalah sebuah struktur dalam pembuatan perangkat lunak. Kelas merupakan bentuk struktur pada kode program yang menggunakan metodologi berorientasi objek.

2. Objek (*object*)

   Objek adalah abstraksi dari sesuatu yang mewakili dunia nyata benda, manusia, satua organisasi, tempat, kejadian, struktur, status, atau hal-hal lain yang bersifat abstra. Objek merupakan suatu entitas yang mampu menyimpan informasi (status) dan mempunyai operasi (kelakuan) yang dapat diterapkan atau dapat berpengaruh pada status objeknya. Objek mempunyai siklus hidup yaitu diciptakan, dimanipulasi, dan dihancurkan. Secara teknis, sebuah kelas saat program dieksekusi makan akan dibuat sebuah objek. Objek dilihat darisegi teknis adalah elemen pada saat *runtime* yang akan diciptakan, dimanipulasi, dan dihancurkan saat eksekusi sehinga sebuah objek hanya ada saat sebuah program dieksekusi.

3. Metode (*method*)

   Operasi atau metode pada sebuah kelas hampir sama dengan fungsi atau prosedur pada metodologi struktural. Sebuah kelas boleh memiliki lebih dari satu metode atau operasi. Metode atau operasi yang berfungsi untuk memanipulasi objek itu sendiri. Operasi atau metode merupakan fungsi atau transformasi yang dapat dilakukan terhadap objek atau dilakukan oleh objek.

   Metode atau operasi dapat berasal dari *event*, aktifitas atau aksi keadaan, fungsi, atau kelakuan dunnia nyata. Contoh metode atau operasi misalnya *Read*, *Write*, *Move, Copy,* dan sebagainya.

4. Atribut (*attribute*)

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

5. Abstraksi (*abstraction*)

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

6. Enkapsulasi (*encapsulation*)

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

7. Pewarisan (*inheritance*)

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

8. Antarmuka (*interface*)

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

9. *Reusability*

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

10. Generalisasi dan Spesialisasi

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

11. Komunikasi Antarobjek

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

12. Polimorfisme (*polymorphism*)

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

13. *Package*

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

2.2.8.2 Pengenalan UML

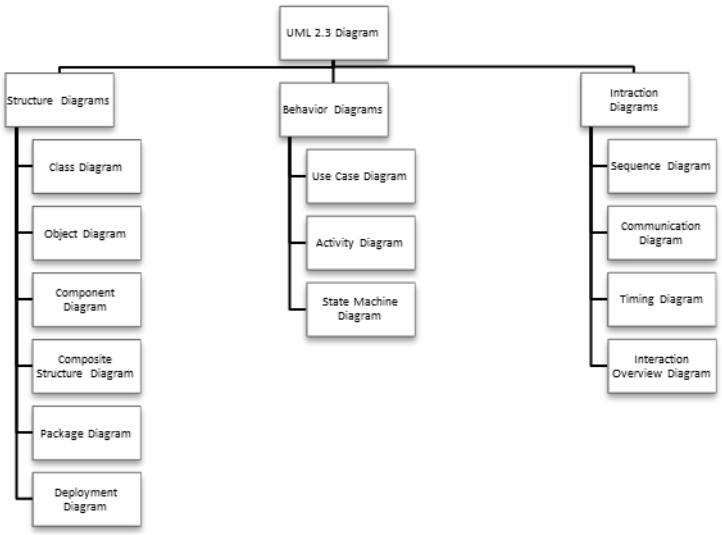
Pada perkembangan teknologi perangkat lunak, diperlukan adanya bahasa yang digunakan untuk memodelkan perangkat lunak yang akan dibuat dan perlu adanya standarisasi agar orang di berbagai negara dapat mengerti pemodelan perangkat lunak. Seperti yang kita ketahui bahwa menyatukan banyak kepala untuk menceritakan sebuah ide dengan tujuan untuk memahami hal yang tidaklah mudah, oleh karena itu diperlukan sebuah bahasa pemodelan perangkat lunak yang dapat dimengerti oleh banyak orang. Banyak orang yang telah membuat bahasa pemodelan pembangunan perangkat lunak sesuai dengan teknologi pemrograman yang berkembang pada saat itu, misalnya yang sempat berkembang dan digunakan banyak pihak adalah *Data Flow Diagram* (DFD) untuk memodelkan perangkat lunak yang menggunakan pemrograman prosedural atau struktural, kemudian juga ada *State Transition Diagram* (STD) yang digunakan untuk memodelkan sistem *real time* (waktu nyata). Pada perkembangan teknik pemrograman berorientasi objek, munculah sebuah standarisasi bahasa pemodelan untuk pembangunan perangkat lunak yang dibangun dengan menggunakan teknik pemrograman berorientasi objek, yaitu *Unified Modeling Language* (UML). UML muncul karena adanya kebutuhan pemodelan voisual untu menspesifikasikan, menggambarkan, membangun, dan dokumentasi dari sistem perangkat lunak.

UML merupakan bahasa visual untuk pemodelan dan komunikasi mengenai sebuah sistem dengan menggunakan digaram dan teks-teks pendukung. UML hanya berfungsi untuk melakukan pemodelan. Jadi penggunaan UML tidak terbatas pada metodologi tertentu, meskipun pada

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

2.2.8.3 Diagram UML

Pada UML 2.3 terdiri dari 13 macam diagram yang dikelompokkan dalam 3 kategori. Pembagian kategori dan macam-macam diagram tersebut dapat dilihat pada gambar 2.2.



Gambar 2.2 Diagram UML[8]

Berikut ini penjelasan singkat dari pembagian kategori tersebut :

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

2.2.8.4 Use Case Diagram

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

Syarat penamaan pada *Use Case* adalah nama didefinisikan sesimpel mungkin dan dapat dipahami. Ada dua hal utama pada *Use Case* yaitu pendefinisian apa yang disebut aktor dan *Use Case.*

2.2.8.5 Activity Diagram

Diagram aktivitas atau *activity diagram* menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis. Yang perlu diperhatikan disini adalah bahwa diagram aktivitas menggambarkan aktivitas sistem bukan apa yang dilakukan aktor, jadi aktivitas yang dapat dilakukan oleh sistem.

Diagram aktivitas juga banyak digunakan untuk mendefinisikan hal-hal berikut :

1. Rancangan proses bisnis dimana setiap urutan aktivitas yang digambarkan merupakan proses bisnis sistem yang didefinisikan

2. Urutan atau pengelompokan tampilan dari sistem/*user interface* dimana setiap aktivitas dianggap memiliki sebuah rancangan antarmuka tampilan
3. Rancangan pengujian dimana setiap aktivitas dianggap memerlukan sebuah pengujian yang perlu didefinisikan kasus ujinya

2.2.8.6 Sequence Diagram

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

Banyaknya diagram sequence yang harus digambar adalah sebanyak pendefinisian *use case* yang memiliki proses sendiri atau yang penting semua *use case* yang telah didefinisikan interaksi jalannya pesan sudah dicakup pada diagram sequence sehingga semakin banyak *use case* yang didefinisikan maka diagram sequence yang harus dibuat juga semakin banyak.

Penomoran pesan berdasarkan urutan interaksi pesan. Penggambaran letak pesan harus berurutan, pesan yang lebih atas dari lainnya adalah pesan yang berjalan terlebih dahulu.

2.2.8.7 Class Diagram

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

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

1) Kelas Main

   Kelas yang memiliki fungsi awal dieksekusi ketika sistem dijalankan.

2) Kelas yang menangani tampilan sistem

Kelas yang mendefinisikan dan mengatur tampilan ke pemakai.

3) Kelas yang diambil dari pendefinisian *usecase*

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

4) Kelas yang diambil dari pendefinisian data

Kelas yang digunakan untuk memegang atau membungkus data menjadi sebuah kesatuan yang diambil maupun akan disimpan ke basis data. Jenis-jenis kelas diatas juga dapat digabungkan satu sama lain sesuai dengan pertimbangan yang dianggap baik asalkan fungsi-fungsi yang sebaiknya ada pada struktur kelas tetap ada. Susunan kelas juga dapat ditambahkan kelas utilitas seperti koneksi ke basis data, membaca *file* teks, dan lain sebagainya sesuai kebutuhan.

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

1. Generalization dan Inheritence

   Diperlukan untuk memperlihatkan hubungan pewarisan (inheritance) antar unsur dalam diagram kelas. Pewarisan memungkinkan suatu kelas mewarisi semua atribut, operasi, relasi, dari kelas yang berada dalam hirarki pewarisannya.

2. Associations

   Hubungan statis antar class. Umumnya menggambarkan class yang memiliki atribut berupa class lain, atau class yang harus mengetahui ekstensi class lain. Dalam notasi UML kita mengenal asosiasi 2 arah (bidirectional) dan 1 arah (undirectional).

3. Aggregation

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

4. Composition

   Aggregation dengan ikatan yang lebih kuat. Di dalam composite aggregation, siklus hidup part class sangat bergantung pada whole class sehingga bila objek instance dari whole class dihapus maka objek instance dari part calss juga akan terhapus.

5. Depedency

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

6. Realization

   Hubungan antar-class dimana sebuah class memiliki keharusan untuk mengikuti aturan yang ditetapkan class lainnya. Biasanya realization digunakan untk menspesifikasikan hubungan antara sebuah interface dengan class yang mengimplementasikan interface tersebut.

2.2.9 Rekayasa Perangkat Lunak

Rekayasa perangkat lunak (*software engineering*) merupakan pembangunan dengan menggunakan prinsip atau konsep rekayasa dengan tujuan menghasilkan perangkat lunak yang bernilai ekonomi, dipercaya dan bekerja secara efisien menggunakan mesin. Rekayasa perangkat lunak lebih fokus pada praktik pengembangan perangkat lunak dan mengirimkan perangkat lunak yang bermanfaat bagi pengguna. Rekayasa perangkat lunak fokus pada bagaimana membangun perangkat lunak yang memenuhi kriteria berikut:

1. Dapat terus dipelihara setelah perangkat lunak selesai dibuat seiiring berkembangnya teknologi dan lingkungan.
2. Dapat diandalkan dengan proses bisnis yang dijalankan dan perubahan yang terjadi.

3. Efisien dari segi sumber daya dan penggunaan.
4. Kemampuan untuk dipakai sesuai dengan kebutuhan.

Pekerjaan yang terkait dengan rekayasa perangkat lunak dapat dikategorikan menjadi tiga buah kategori umum tanpa melihat area dari aplikasi, ukuran proyek atau kompleksitas perangkat lunak yang akan dibangun, yaitu:

a. Fase Pendefinisian (*Definition Phase*)

   Fase pendefinisian fokus pada "*what*" yang artinya harus mencari tahu atau mengidentifikasi informasi apa yang harus diproses, seperti apa fungsi dan performansi yang diinginkan, seperti apa prilaku sistem yang diinginkan, apa kriteria validasi yang dibutuhkan untuk mendefinisikan sistem.

b. Fase Pengembangan (*Development Phase*)

   Fase pengembangan fokus pada "*how*" yang artinya selama tahap pengembangan perangkat lunak seorang perekayasa perangkat lunak (*software engineer*) berusaha untuk mendefinisikan bagaimana data diinstrukturkan dan bagaimana fungsi-fungsi yang dibutuhkan diimplementasikan, bagaimana karakter antarmuka tampilan, bagaimana desanin ditranslasikan ke dalam bahasa pemrograman dan bagaimana pengujian dijalankan.

c. Fase Pendukung (*Support Phase*)

   Fase pendukung fokus pada perubahan yang terasosiasi pada perbaikan kesalahan (*error*), adaptasi yang dibutuhkan pada lingkungan perangkat lunak yang terlibat dan perbaikan yang terjadi akibat perubahan kebutuhan pengguna. Fase pendukung terdiri dari empat tipe perubahan antara lain:

1. Koreksi (*Correction*), yaitu pemeliharaan dengan melakukan perbaikan terhadap kecacatan perangkat lunak.
2. Adaptasi (*Adaptation*), yaitu merupakan tahap untuk memodifikasi perangkat lunak guna mengakomodasi perubahan lingkungan luar dimana perangkat lunak dijalankan.

3. Perbaikan (*Enchancement*), pemeliharaan perfektif atau penyempurnaan melakukan eksekusi atau penambahan pada kebutuhan fungsional sebelumnya.
4. Pencegahan (*Prevention*), pencegahan atau sering disebut juga dengan rekayasa ulang sistem (*software reengineering*) harus dikondisikan untuk mempu melayani kebutuhan yang diinginkan pemakainya.

2.2.10 Perangkat Lunak Pendukung

Adapun beberapa perangkat lunak pendukung dalam penyusunan Laporan sistem monitoring menggunakan teknologi RFID Untuk Pembangunan Perangkat Lunak otomasi Bike.Bdg adalah

2.2.10.1 MYSQL

MySQL adalah sebuah produk *Relational Database Management System* (RDBMS)[8]. MySQL dapat mengakses data, memanipulasi data, melindungi data dari korupsi dan inkonsistensi serta mempertahankan metadata yang diperlukan untuk menentukan data yang disimpan dalam *database*. Perbedaan antara DBMS dengan RDBMS adalah bukan hanya dari mendukung atau tidaknya penyimpanan data dalam tabel, tetapi juga hubungan antartabel. MySQL menyediakan banyak fitur yang mendukung sebuah lingkungan yang aman untuk menyimpan, memelihara, dan mengakses data. .NET Framewok Bahasa Pemrograman Visual Basic.Net.

2.2.10.2 .NET Framework

.NET Framework adalah *platform* yang merupakan perwujudan teknologi modern. Framework .NET diciptakan untuk dapat memecahkan masalah yang banyak dihadapi dunia pemrograman secara lebih efisien. Metodologi perancangan berorientasi objek dan konsep software sebagai komponen tertanam dengan kuat di Framework.NET.

2.2.10.3 Bahasa Pemrograman Visual Basic.Net

Visual Basic.Net merupakan salah satu *tool development Microsoft* yang dapat digunakan untuk membuat aplikasi di lingkungan kerja berbasis sistem operasi windows. Visual Basic.Net menyediakan tools bagi para developer untuk membangun aplikasi yang berjalan di .Net framework. Visual Basic.Net membawa perubahan besar dalam gaya pemrograman, karena setiap programer di tuntut untuk memahami .NET object model dan object oriented programming dengan baik, jika tidak ingin menghasilkan aplikasi dengan performa rendah. Visual Basic.Net juga semakin mempertipis jarak antara Windows Programmer dengan Web Programmer[9].

2.2.10.4 Visual Studio

Microsoft Visual Studio merupakan sebuah perangkat lunak yang dapat digunakan untuk melakukan pengembangan aplikasi, baik aplikasi bisnis, aplikasi personal, ataupun komponen aplikasinya dalam bentuk aplikasi console, aplikasi Windows, ataupun aplikasi Web. Visual Studio mencakup kompiler, SDK, *Integrated Development Environment* (IDE), dan *Library*. Kompiler yang dimasukkan ke dalam paket Visual Studio antara lain Visual C++, C#, VB.NET, J++ dan J#.

Microsoft Visual Studio dapat digunakan untuk mengembangkan aplikasi dalam *unmanaged*/*native code* (dalam bentuk bahasa mesin yang berjalan di atas Windows) ataupun *managed* code (dalam bentuk Microsoft Intermediate Language di atas .NET Framework). Selain itu, Visual Studio juga dapat digunakan untuk mengembangkan aplikasi Silverlight, aplikasi Windows Mobile (yang berjalan di atas .NET Compact Framework).

Beberapa keuntungan Visual Studio:

1. Dapat membuat aplikasi untuk beberapa platform termasuk Windows 7,Windows Server 2008 R2, SQL Server 2008, SharePoint. Office, Windows Azure, dan Windows Phone.
2. Lingkungan pemrograman dapat dipersonalisasi

3. Dukungan *Application Lifecycle Management* dapat memudahkan kolaborasi, melakukan *testing* ataupun *debugging*.

2.2.10.5 Web Service

Web service adalah sekumpulan *application logic* beserta objek-objek dan metode-metode yang dimilikinya, yang terletak di suatu server yang terhubung ke internet sehingga dapat diakses menggunakan *protocol* HTTP dan SOAP (*Simple Object Access Protocl*). Dalam penggunaanya, *web service* dapat digunakan hanya untuk memeriksa data *user* yang login ke sebuah website ataupun untuk digunakan pada transaksi perbankan online yang rumit.

Tujuan dari teknologi ini adalah untuk memudahkan beberapa aplikasi komponenya untuk saling terhubung dengan aplikasi lain dalam sebuah organisasi maupun diluar organisasi menggunakan standar yang tidak terikat *platform* (platform-*neutral*) dan tidak terikat akan bahasa pemrograman yang digunakan. Hal tersebut dapat terjadi karena XML standar yang didukung oleh banyak perusahaan besar didunia, yang digunakan untuk bertukar data. Selain daripada itu, penggunaan SOAP menjadi metode-metode dari objek-objek yang ada dalam sebuah *web service* dapat di akses dari aplikasi lain seperti halnya aplikasi tersebut mengakses metode *local* [12].

2.2.11 Radio Frequency Identification (RFID)

RFID (Radio Frequency Identification) adalah sebuah metode identifikasi dengan menggunakan sarana yang disebut label RFID atau transponder (t*ag*) untuk menyimpan dan mengambil data jarak jauh. Label atau kartu RFID adalah sebuah benda yang bisa dipasang atau dimasukkan di dalam sebuah produk, hewan atau bahkan manusia dengan tujuan untuk identifikasi menggunakan gelombang radio[7].

RFID merupakan sebuah teknologi yang menggunakan media radio frekuensi dalam proses identifikasinya. Identifikasi dilakukan secara otomatis terhadap objek-objek atau manusia tanpa memerlukan operasi manual. Maka dapat disimpulkan, RFID adalah teknologi penangkapan data yang dapat

digunakan secara elektronik untuk mengidentifikasi, melacak dan menyimpan informasi yang tersimpan dalam t*ag* RFID[11].

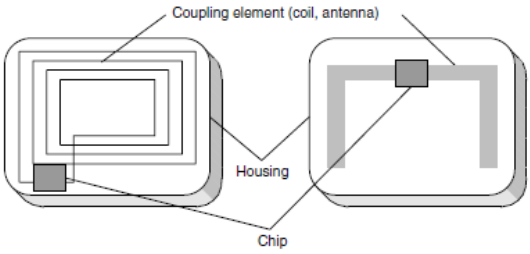
2.2.11.1 Komponen Utama RFID

RFID terdiri atas 3 komponen utama, yaitu:

1. RFID t*ag*, berisi antena sebagai *coupling element* yang memungkinkan untuk menerima dan merespon terhadap suatu query yang dipancarkan oleh suatu RFID *reader* . Juga terdapat chip yang mampu menyimpan sejumlah informasi unik.
2. *Reader* RFID, berisi modul frekuensi radio (*transmitter* dan *receiver),* pengontrol dan *coupling element* ke t*ag* .
3. *Host computer*, terdiri dari atas Basis data yang menyimpan semua data yang terkandung dalam t*ag.* Sistem computer yang mengatur alur informasi dari item-item yang terdeteksi dalam lingkup sistem RFID dan mengatur komunikasi antara *tag* dan *reader* .

2.2.11.1.1 RFID Tag

RFID *tag* adalah sebuah benda kecil, komponen yang terdiri dari chip dan antenna. *Tag* ini dapat berupa stiker adesif yang ditempelkan pada suatu barang atau produk. Selain itu, *tag* ini juga dapat berupa koin dan kartu. Bentuk dan struktur dari *tag* ini dibuat fleksible sesuai dengan objek yang akan diidentifikasi. Gambar 2.3 menunjukkan *layout* dasar dari sebuah RFID *tag :*



Gambar 2.3 Layout Dasar RFID Tag [11]

RFID *tag* ini akan bekerja ketika menerima sinyal dari RFID *reader* dan akan memantulkan kembali sinyal tersebut. Ada 3 jenis RFID *tag* berdasarkan sifatnya, yaitu :

1. *Passive* RFID t*ag*

Passive RFID t*ag* adalah *tag* yang bersifat pasif karena tidak memiliki power supply sendiri. *Tag* ini bekerja dengan berbekal induksi listrik yang ada pada antenna kecil yang terkandug didalamnya. Bila ada frekuensi radio scanning yang masuk dari *reader* barulah induksi listrik akan terjadi sehingga t*ag* dapat merespon dengan mengirimkan data (informasi unik) yang disimpannya. Biasanya informasi sederhana seperti hanya nomor ID saja.

Dengan tidak adanya power supply pada t*ag* pasif maka ukurannya bisa dibuat sekecil mungkin. Saat ini RFID t*ag* yang sudah beredar di pasaran ada yang bisa diletakkan di bawah kulit. Namun jarak jangkauan dimana *tag* ini dapat terdeteksi oleh *reader* sangat terbatas (beberapa cm). Gambar 2.4 dibawah ini adalah contoh dari *Tag* pasif dengan spesifikasi sebagai berikut:

1) Frekuensi kerja : 125 kHz
2) Jarak pembacaan : 8-14 cm
3) Format : GK4001
4) Data (64-bit / dll) : 64-bit



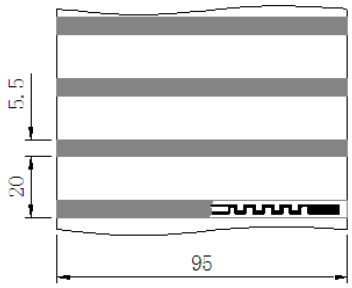
Gambar 2.4 RFID Tag Pasif

2. *Active* RFID *Tag*

Active RFID *tag* memiliki power supply sendiri, misalnya baterai yang dapat digunakan sebagian atau seluruhnya untuk mengaktifkan *circuit microchip* dan antenna, juga untuk mengirimkan sinyal ke *reader* . Beberapa *tag* juga dapat dihubungkan dengan catu daya dari luar. *Tag* ini memiliki jarak jangkauan yang lebih jauh. Memori yang dimilikinya juga lebih besar sehingga bisa menampung berbagai macam informasi di dalamnya. Jarak jangkauan dari RFID *tag* yang aktif ini bisa sampai sekitar 10 meter dan dengan umur baterai yang bisa mencapai

beberapa tahun lamanya. Dibawah ini adalah contoh dari *tag* aktif dengan spesifikasi sebagai berikut:

a. Karakteristik Fisik



Gambar 2.5 Karakrteristik Fisik RFID Tag Aktif

1) Ukuran : 95 × 5.5 × 0.28mm (3,7 × 0,22 × 0.01in)
2) Bahan Dasar : PET; alumunium
3) Bahan Casing : Perekat
4) Diameter Bagian Inti : 76.2 mm (3in)
5) Inti Lebar : 95 mm (3.7in)
6) Jumlah satu paket : 5.000 inlays per roll; sampai 12 gulungan per karton
7) Diameter Gulungan : 180 ± 5mm (7,1 ± 0.2in)
8) Berat : 2.32kg ± 0.05kg/roll; 10.0kg ± 0.2kg/carton
   (5,1 ± 0.11lbs/roll; 22,0 ± 0.44lbs/carton)

b. Spesifikasi Kinerja

1) Frekuensi Kerja : 860-960Mhz
2) Standar yang didukung : EPCglobal Kelas 1 Gen 2; ISO 18000-6C
3) Jarak Baca : ± 6 meter / 19.7 kaki (tergantung *reader*)
4) Jarak Tulis : ± 4 meter / 13.1 kaki (tergantung *reader*)
5) Jenis Chip : UCODE G2XM
6) Memori : 240-bit EPC; 64-bit TID;
   512-bit user memory
7) Fungsi : Read/Write
8) Data Retensi : 50 tahun
9) Daya Tahan Tulis : 100.000 siklus
10) Anti-collision : Operasi beberapa *Tag*



Gambar 2.6 RFID Tag Aktif XCTF-8102A

3. *Semipasif* RFID *Tag*

Tag semipasif menyerupai t*ag* aktif dimana *tag* jenis ini juga mempunyai catu daya sendiri, namun baterai yang dimiliki hanya digunakan untuk mengaktifkan *microchip* dan antenna, tidak digunakan untuk mengirimkan sinyal ke *reader* . Untuk proses *sinyal broadcast*, metodenya sama dengan cara *tag* pasif memantulkan sinyal ke *reader* .

Ketika mengaktifkan *tag*, catu daya yang dipakai adalah catu daya sendiri, sehingga tidak dipengaruhi oleh besarnya daya yang di induksikan oleh *reader* . Tetapi, saat pengiriman balik, daya yang digunakan sepenuhnya berasal dari *reader*, sehingga akan terjadi kegagalan kalau jaraknya terlalu jauh. Berikut adalah Tabel yang merupakan perbadingan spesifikasi beberapa jenis *tag* .

Tabel 2.7 Perbandingan beberapa jenis *Tag*

	Tag Pasif	*Tag* Semipasif	*Tag* Aktif
Catu daya	Eksternal (dari *reader*)	Baterai internal	Baterai internal
Rentang baca	Dapat mencapai 6 meter	Dapat mencapai 30 meter	Dapat mencapai 228 meter
Tipe Memori	Umumnya read-only	Read-write	Read-write
Harga	$0.2 hingga beberapa dollar	$2 hingga $10	$20 atau lebih
Usia *Tag*	Dapat mencapai 20 tahun	2 sampai 7 tahun	5 sampai 10 tahun

Berdasarkan tipe memori yang dimiliki oleh t*ag* , maka terdapat 3 variasi *tag :*

1. *Tag read-only*, *tag* ini memiliki kapasitas memori yang kecil (biasanya kurang ari 64 bit) dan data yang terprogram didalamnya bersifat permanen sehingga data tidak dapat diubah. *Tag* ini hanya bisa digunakan sekali, tidak dapat re-use. Informasi yang terkandung didalam *tag* seperti ini

terutama adalah informasi identifikasi item. *Tag* pasif biasanya memiliki tipe memori seperti ini.

2. *Tag read/write,* data dapat di update jika diperlukan. Namun sebagai konsekuensinya kapasitas memorinya akan lebih besar dan harganya lebih mahal dibandingkan t*ag read-only. Tag* seperti ini biasanya digunakan bila data yang tersimpan didalamnya perlu untuk di di update seiring dengan daur hidup produk, misalnya di pabrik.
3. *Tag write-once read many,* informasi disimpan sekali dan tidak bisa dilakukan perubahan berikutnya terhadap data tapi dapat dibaca berulang-ulang. *Tag* tipe ini memiliki fitur keamanan *read-only* dengan menambahkan fungsionalitas tambahan dari *Tag read/write.*

2.2.11.1.2 RFID Reader

Prinsip kerja RFID *reader* serupa dengan *tranceiver* radio, yaitu memancarkan dan menerima. *Reader* ini dalam kondisi siaga akan memancarkan gelombang elektromagnetik sesuai dengan daya jangkaunya. Ketika ada *tag* memasuki area jangkauannnya, t*ag* akan mendapat daya dari gelombang elektromagnet *reader* .

Dari daya yang diperoleh, *tag* memancarkan data yang dibawa. Data pancaran tersebut akan diterima oleh *reader* . Selanjutnya data yang diterima tadi akan diteruskan pada aplikasi untuk diolah sesuai dengan rancangan sistem.

Dalam fungsinya, RFID *reader* dituntut untuk dapat melakukan dua tugas, yaitu berkomunikasi melalui gelombang radio dan membaca data yang dibawa *tag* kemudian diteruskan ke aplikasi.



Gambar 2.8 RFIDReader RapidACCESS RFD220P[11]

Gambar diatas adalah jenis *reader* RFID RapidACCESS RFD220P, RapidACCESS RFD220P adalah pembaca tetap / desktop terintegrasi yang dapat membaca dan menulis label elektronik atau yang dilengkapi dengan t*ag* ISO-18000-6B standar, EPC Kelas 1 standar G2 dan standar EM-4222. Berkisar membaca maksimal hingga 4 meter (tergantung t*ag*). Spesifikasinya yaitu sebagai berikut:

1. Read-Write *tag* : *tag* elektronik dengan ISO-18000-6B, EPC Class1 G2 danstandar EM-4222 .
2. Mendukung komunikasi data melalui port RS232, Wiegand pelabuhan, pelabuhan RS485
3. Modus kerja: Frekuensi Hopping Spread Spectrum (FHSS)
4. Max. RF output daya: 20 ~ 30dBm
5. Kecepatan membaca: merata, 6ms / 32bits.
6. Menulis kecepatan: merata, 50ms/32bits.
7. Mekanisme keamanan: Verifikasi keabsahan *Tag* .
8. Built-in anti-tabrakan: Baca hingga 100 *Tag* pada suatu waktu.
9. 3 (tiga) Mode Operasi Tersedia:
    a. CONTINOUS: *Tag* otomatis membaca secara berkala
    b. TRIGGER: Kontrol membaca *Tag* dengan memasukkan memicu sinyal
    c. COMMAND: Memulai membaca / menulis dengan perintah software / API

10. Power supply: 110V ~ 220V eksternal VAC auto mulai.
11. Dimensi: 250mm x 250mm x 77mm
12. Berat: 0.9kg (tanpa kabel dan bracket)
13. Suhu operasi: -20 C sampai 55 C
14. Jarak membaca *Tag* hingga 4 meter, menulis *Tag* hingga 1,5 meter (tergantung *Tag*)

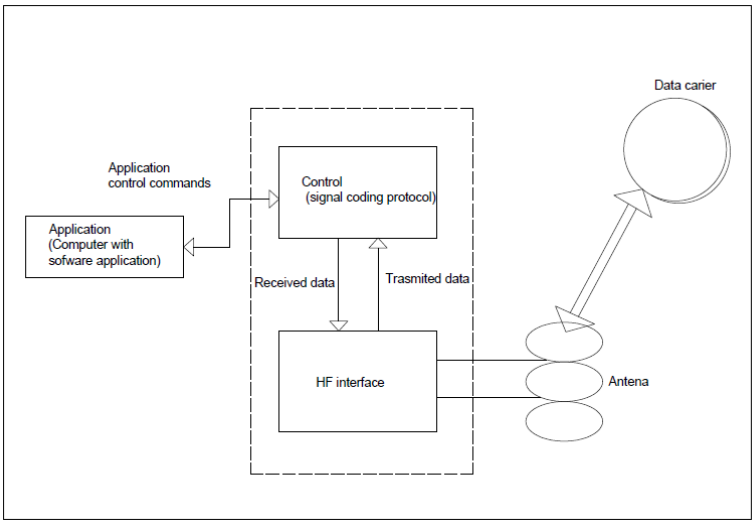


Gambar 2.9 RFID Reader ID-12 Innovations

Gambar diatas adalah salah satu jenis *reader* RFID jenis ID-12 Innovations. RFID tersebut merupakan sarana pengembangan RFID berbasis *reader* tipe ID-12 yang telah dilengkapi jalur komunikasi RS-232 sehingga dapat dihubungkan ke COM port komputer secara langsung. Modul ini cocok untuk mengembangkan aplikasi-aplikasi berbasis RFID, misalnya: mesin absensi RFID, RFID access controller, dsb. Spesifikasinya adalah sebagai berikut:

1) Catu daya : 5VDC
2) Frekuensi kerja : 125 kHz
3) Jarak pembacaan : ± 12 cm
4) Antarmuka : UART TTL / RS-232), Wiegand26, maupun Magnetic ABA Track 2
5) Format kartu : EM4001
6) Fitur : Modul ID-12 dapat diganti dengan modul ID-Series yang lain.
7) Dimensi : 8.1 cm (p) x 6.2 cm (l) x 1.8 cm (t)

Reader dibedakan berdasarkan kapasitas penyimpanannya, kemampuan pemrosesannya, serta frekuensi yang dapat dibacanya. Kompabilitas *reader* terhadap t*ag* juga perlu diperhatikan karena tidak semua t*ag* dengan *reader* akan *match,* ada pasangannya masing-masing. Berikut ini gambar 2.14 adalah bagan 2 blok fungi dari *reader*



Gambar 2.10 Bagan 2 Blok Fungsi dari Reader [11]

a. HF Interface

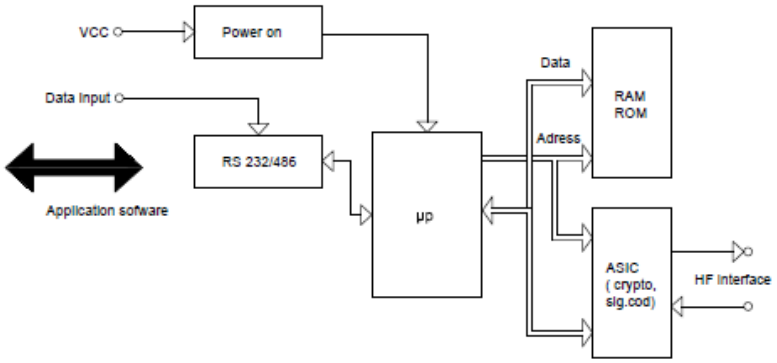
High Frequency Interface digunakan oleh *reader* untuk menghasilkan sinyal transmisi agar dapat mengaktifkan *tag* dan menyuplai daya (pada *tag* pasif), memodulasi sinyal transmisi untuk mengirimkan data ke *tag* , serta menerima dan mendemodulasikan sinyal transmisi dari sebuah *tag* .

HF Interface terdiri dari dua jalur sinyal yang terpisah, berdasarkan pada dua arah aliran data dari dan ke *tag* , Data yang dikirimkan ke *tag* melalui bagian transmisi dan data yang diterima dari t*ag* akan di proses pada bagian *receiver*.

b. *Control Sistem*

Gambar 2.15 menunjukkan bagian *control* sistem dari sebuah *reader* . *Control* sistem ini memiliki fungsi-fungsi sebagai berikut:

1. Berkomunikasi dengan aplikasi perangkat lunak dan melakukan perintah dari aplikasi perangkat lunak.
2. Mengontrol metode komunikasi dengan *tag*
3. Melakukan proses *coding* dan *decoding* dari sinyal informasi



Gambar 2.11 Diagram blok dari conrol unit [11]

Control unit memiliki sistem sesuai dengan mikroprosessor yang digunakan. Pertukaran data antara aplikasi perangkat lunak dan *reader* dilakukan menggunakan komunikasi serial melalui RS232 *interface*.

2.2.11.2 Sistem Pada Beberapa Kategori RFID

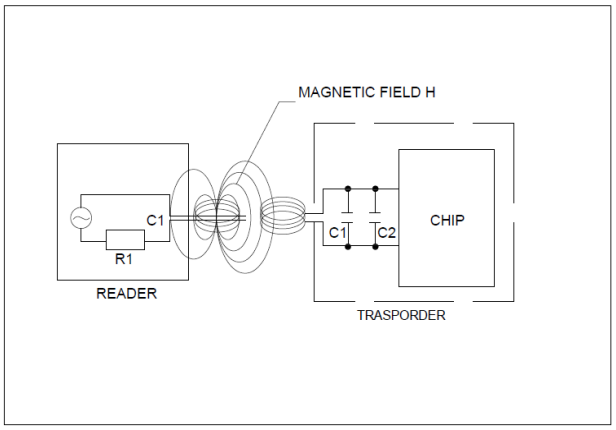
Sistem pada RFID dibagi menjadi 3 kategori berdasarkan sifat *tag* yang digunakan, yaitu sistem pada RFID pasif, sistem pada RFID aktif dan sistem pada RFID semipasif.

2.2.11.2.1 Sistem Pada RFID Pasif

Secara umum, proses yang terjadi saat sistem RFID pasif beroperasi adalah:

1. *Tag* memasuki medan elektromagnetik frekuensi radio
2. Sinyal RF mensuplai energi ke t*ag*
3. *Tag* mentransmisikan ID dan data yang tersimpan didalamnya
4. *Reader* menangkap data yang dikirim oleh *tag*
5. *Reader* mengirim data dari *tag* tersebut ke komputer
6. Komputer mengolah dan menganalisis data
7. Komputer memberikan instruksi kepada *reader*
8. *Reader* mentransmisikan data ke *tag*

Gambar 2.16 adalah ilustrasi dari proses yang terjadi pada sistem RFID pasif:



Gambar 2.12 Proses Catu Daya pada Sistem RFID Pasif [11]

Inductive coupled tag yang beroperasi secara pasif, terdiri dari *microchip* tunggal dan area kumparan yang luas sebagai antena. *Reader* yang menjadi pensuplai daya untuk mengaktifkan microchip, dimana antena *reader* akan membangkitkan medan elektromagnetik (dengan frekuensi tinggi) yang kuat dan akan menembus area kumparan dan di sekitar kumparan.

Sebagian medan akan menembus kumparan *tag* yang letaknya berjauhan dengan *reader* dan menghasilkan tegangan pada antena *tag* akibat proses induksi. Tegangan ini akan diserahkan oleh dioda pada rangkaian *tag* dan kemudian digunakan sebagai sumber tegangan untuk mengkatifkan chip.

Kapasitor dihubungkan secara paralel dengan kumparan antena untuk menghasilkan frekuensi resonansi yang sesuai dengan frekuensi *reader*. Kumparan antena dan kapasitor ini telah disesuaikan dengan frekuensi *reader*, dan tegangan induksi akan mencapai titik maksimum saat resonansi meningkat pada rangkaian paralel tersebut.

Pada *inductive coupled sistem* proses transmisi sinyal antara kedua perangkat tersebut tergantung pada kumparan yang dimiliki pada antena keduanya. Sinyal *feedback* dari *tag* ke antena *reader* dapat direpresentasikan sebagai impedansi pada kumparan antena *reader* . Perubahan resistor beban dari *on* dan *off* akan mengubah impedansi ini dan juga mengubah tegangan pada antena *reader* .

Perubahan resistor ini diatur oleh data, sehingga data dapat dikirimkan dari t*ag* ke *reader* , pengiriman ini disebut *load modulation.* Sinyal yang diterima oleh

reader akan dimodulasi dengan cara menyerahkan tegangan yang masuk ke *reader*. Sehingga hasil modulasinya berupa amplitudo dari sinyal modulasi.

2.2.11.2.2 Sistem pada RFID semi pasif

Sistem yang ketiga adalah penggabungan sifat dan atribut dari sistem RFID pasif dan aktif. Sistem RFID semipasif ini mempunyai arsitektur yang cukup unik karena menggunakan baterai sendiri untuk sumber power dalam memproses data internal sedangkan untuk meradiasikan data kembali ke *reader* menggunakan power dari sinyal yang diterima dari *reader*