BAB III

OBJEK DAN METODE PENELITIAN

3.1 Objek Penelitian

Objek penelitian ini adalah Aplikasi Sistem Informasi Penjualan dan Pembelian Barang Menggunakan Barcode Berbasis Intranet pada CV. Kurnia Jaya Mandiri Bandung. Beralamat di Jalan Stasiun Lama Kiaracondong No. 17 A Kel. Kebon Jayanti Kec. Kiaracondong Kota. Bandung Barat, Telp 022-7316752.

3.1.1 Sejarah Singkat Perusahaan

CV. Kurnia Jaya Mandiri merupakan perusahaan kecil yang bergerak di bidang bisnis penjualan yaitu menjual kebutuhan sehari-hari untuk melayani kebutuhan instan di tempat umum seperti di stasiun kiaracondong, pasar dan masyarakat sekelilingnya baik eceran, retail dan grosir. CV. Kurnia Jaya Mandiri berdiri berawal dari Pak H. Suraji beserta keluarga yang ingin mempunyai usaha mandiri di bidang bisnis penjualan yang memanfaatkan tempat yang strategis di area Stasiun Kiaracondong untuk memberikan pelayanan jasa kebutuhan sehari-hari. Perusahaan ini didirikan pada bulan Agustus tahun 2000 di Kiaracondong.

Perekrutan pegawai pertama kali dilakukan dengan cara memberikan pelatihan langsung kelapangan. Dengan mencari order dari toko ke toko, mengirim barang kemudian dibuatkan struktur organisasi untuk lebih memudahkan dalam hubungan wewenang dan tanggung jawab. Bagian yang ada di CV. Kurnia Jaya Mandiri adalah

Bagian Penjualan, Bagian Keuangan, dan Bagian Pengiriman yang diawasi oleh seorang Pimpinan Perusahaan.

3.1.2 Visi dan Misi Perusahaan

CV. Kurnia Jaya Mandiri memiliki Visi dan Misi, yaitu:

3.1.2.1 Visi Perusahaan

Visi dari CV. Kurnia Jaya Mandiri adalah menjadikan kebutuhan sehari-hari masyarakat dapat dengan mudah terpenuhi, dan sekaligus sebagai prospek bisnis yang dapat menguntungkan dalam hal materi dan non materi dengan cara memanfaatkan potensi Sumber daya Alam (SDA) yang tersedia dan Sumber Daya Manusia (SDM) yang ada.

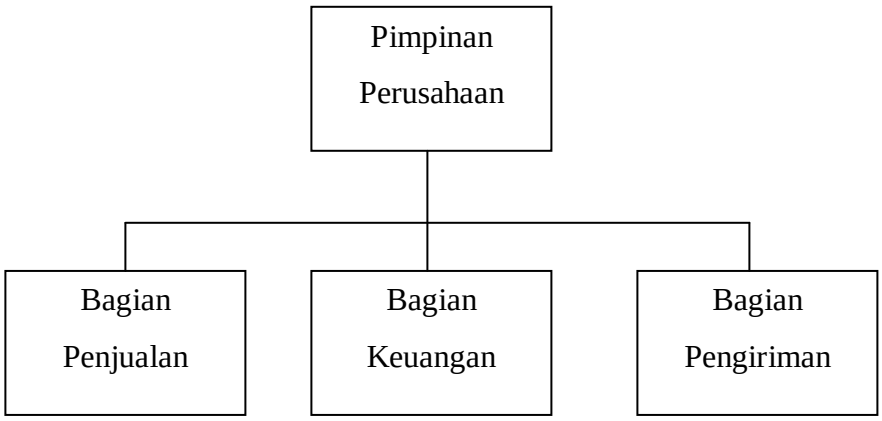
3.1.2.2 Misi Perusahaan

Misi dari CV. Kurnia Jaya Mandiri adalah menjadi perusahaan profesional di bidang bisnis penjualan kebutuhan sehari-hari.

3.1.3 Struktur Organisasi Perusahaan

Setiap perusahaan pasti memiliki struktur organisasi masing-masing yang berbeda antara perusahaan yang satu dengan perusahaan yang lain. Struktur organisasi dibuat agar susunan kerja, wewenang, dan tanggung jawab suatu bagian

tertentu dapat terlihat dengan jelas. Begitu juga dengan CV. Kurnia Jaya Mandiri yang telah memiliki struktur organisasi sebagai berikut :



Gambar 3.1 Struktur Organisasi CV. Kurnia Jaya Mandiri

3.1.4 Deskripsi Tugas

Agar setiap karyawan dapat menjalankan fungsi serta tugasnya masing -masing dengan baik maka perlu dijabarkan secara jelas tentang deskripsi kerjanya masing -masing yang sesuai dengan posisi jabatan yang diambil di dalam perusahaan tersebut. Adapun deskripsi kerja masing-masing bagian yang ada didalam CV. Kurnia Jaya Mandiri adalah sebagai berikut :

1) Pimpinan Perusahaan
    a. Mengawasi semua kegiatan dalam perusahaan.

b. Menerima dan memeriksa kebenaran laporan – laporan dari setiap bagian dalam perusahaan.

c. Melakukan kebijakan perusahaan di bidang penjualan dan pembelian.

d. Mengawasi dan mengevaluasi rencana kerja serta anggaran yang disetujui berdasarkan skala prioritas perusahaan dalam menunjang sasaran dan tujuan perusahaan.

2) Bagian Penjualan

a. Bertanggung jawab mengenai pemasaran dan penjualan produk.

b. Mengatur strategi pemasaran dan mengadakan promosi.

c. Memberikan laporan secara periodik mengenai laporan penjualan barang kepada pimpinan perusahaan.

d. Melaksanakan target penjualan

3) Bagian Keuangan

a. Bertanggung jawab langsung pada perusahaan dalam mengatur administrasi dan segala urusan yang menyangkut keuangan perusahaan.

b. Memperlancar lalu lintas dan distribusi kesegala pihak, baik pihak dalam maupun luar perusahaan.

c. Memberikan laporan secara periodik mengenai laporan keuangan barang kepada pimpinan perusahaan.

d. Mengkordinir dan mengawasi pelaksanaan penyusunan kebutuhan perusahaan.

3.2 Metode Penelitian

3.2.1 Desain Penelitian

Desain atau rancangan penelitian yang penulis lakukan dalam penelitian ini yaitu deskriptif dengan pendekatan Studi kasus pada CV. Kurnia Jaya Mandiri.

3.2.2 Jenis dan Metode Pengumpulan Data

Untuk memperoleh data yang diperlukan dalam penelitian ini, penulis menggunakan teknik pengumpulan data dengan menggunakan sumber data primer dan sumber data sekunder.

3.2.2.1 Sumber Data Primer

Adapun kedua data tersebut adalah sebagai berikut :

a. Wawancara

Wawancara dilakukan dengan pihak-pihak terkait yang berkompeten dengan harapan dapat melengkapi data-data yang diperlukan dalam penelitian ini, dalam hal ini karyawan yang ditunjuk oleh Pimpinan untuk memberikan informasi kepada peneliti yaitu karyawan pada bidang-bidang yang lainnya. Disini penulis mewawancarai dibagian penjualan dan pembelian mengenai bagaimana cara atau proses penjualan barang kepada konsumen atau kepada pelanggan dan bagaimana pula cara menjadi pelanggan baru.

b. Observasi

Peneliti langsung mengunjungi lokasi penelitian ke CV. Kurnia Jaya Mandiri yang dijadikan objek penelitian, dalam hal ini peneliti melakukan observasi pada bagian penjualan yang menangani data penjualan dan pembelian. Setelah itu peneliti mengamati bagaimana cara pembuatan laporan penjualan dan pembelian apakah sudah terkomputerisasi atau belum dan juga mengamati bagaimana mengetahui stok barang yang tersedia apakah masih menghitung manual atau sudah terkomputerisasi.

3.2.2.2 Sumber Data Sekunder

Sumber data sekunder merupakan sumber data yang informasinya kita dapatkan dari media atau sumber-sumber yang telah ada. Pada sumber data sekunder, penulis mengumpulkan data dengan cara melakukan studi pustaka dan data – data fisik dari perusahaan tersebut berupa laporan, faktur, dan struk pembayaran.

3.2.3 Metode Pendekatan & Pengembangan Sistem

Adapun metode pendekatan dan pengembangan sistem yang digunakan adalah sebagai berikut :

3.2.3.1 Metode Pendekatan Sistem

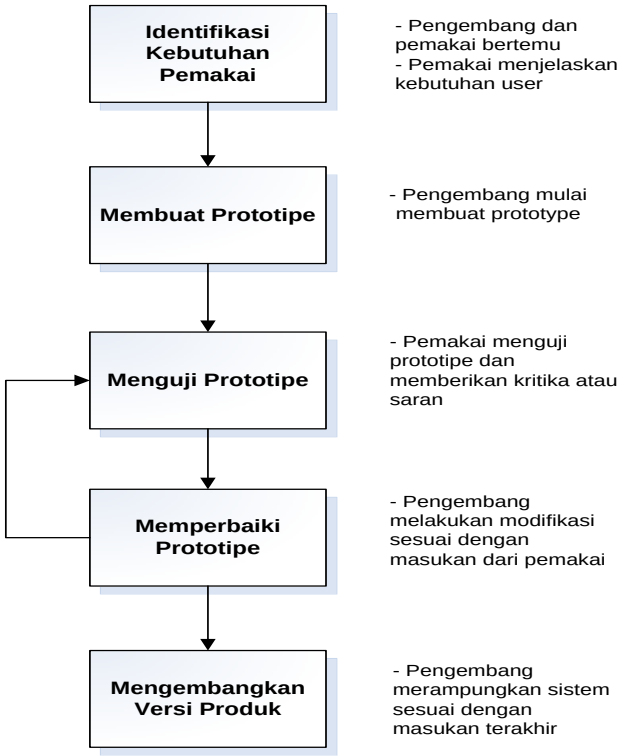
Pendekatan sistem merupakan pendekatan bagaimana menggunakan alat-alat dan peraturan-peraturan yang melengkapi satu atau lebih tahapan-tahapan pengembangan sistem. Adapun metode pendekatan sistem yang digunakan adalah metode terstruktur yang memiliki karakteristik berorientasi pada proses dan data, adapun alat-alat yang digunakan dalam metode pendekatan sistem adalah *Object Oriented* yang di visualisasikan dengan UML, yaitu antara lain : *Use Case, Sequence Diagram, Class Diagram, Collaboration Diagram, Component Diagram dan Deployment Diagram.*

3.2.3.2 Metode Pengembangan Sistem

Dalam memilih metode pengembangan sistem yang efektif penulis menggunakan metode prototype karena metode ini menawarkan bagi pembangun sistem apabila tidak memiliki kepastian terhadap efisiensi algoritma, kemampuan penyesuaian dari sebuah sistem operasi atau bentuk-bentuk yang harus dilakukan oleh interaksi manusia dengan mesin. Dilihat dari situasi tersebut metode prototype menawarkan pendekatan yang terbaik.

Pada awalnya segala hal yang dibutuhkan untuk penelitian dikumpulkan terlebih dahulu. Pembangun sistem dan calon pemakai bertemu langsung untuk mendefinisikan kebutuhan apa saja yang diperlukan untuk perangkat lunak, kriteria perangkat lunak seperti apa yang diinginkan pemakai dan mencari proses-proses yang

masih memerlukan pendefinisian, kemudian dilakukan perancangan kilat terhadap kebutuhan yang telah teridentifikasi pada pertemuan yang biasa disebut dengan pembuatan prototype. Prototype inilah yang akan mengarahkan penulis kepada pembangunan perangkat lunak yang akan dimanfaatkan oleh pemakai. Setelah itu prototype tersebut akan dievaluasi oleh pemakai dan digunakan untuk memperbaiki spesifikasi kebutuhan. Proses ini akan berulang sampai prototype yang dikembangkan memenuhi seluruh kebutuhan pemakai.



Gambar 3.2 Mekanisme Pengembangan Sistem dengan *Prototype* (**Sumber :** Abdul Kadir, Pengenalan Sistem Informasi 2003, Andi. Yogyakarta)

Metodologi yang digunakan dalam perancangan sistem ini yaitu menggunakan model *Prototype*.

Adapun tahapan-tahapan dari metode *prototype* adalah sebagai berikut :

a. Mengidentifikasi kebutuhan pemakai.

   Pada tahap ini analisis sistem akan melakukan studi kelayakan dan studi terhadap kebutuhan pemakai, baik yang meliputi model *interface*, teknik prosedural, maupun dalam teknologi yang akan digunakan.

b. Mengembangkan *prototype*.

   Pada tahap kedua ini, dilakukan kombinasi antara analisa sistem dengan pemrograman dalam mengembangkan *prototype* sistem untuk memperlihatkan kepada pemakai permodelan sistem yang akan dibangunnya.

c. Menguji Sistem

   Menguji prototype yang telah dibangun apakah dapat diterima oleh pemakai.

d. Evaluasi Sistem

   Dalam tahap ini pengembang akan memodifikasi bagian-bagian yang perlu diperbaiki sesuai dengan saran dan masukan pemakai di tahap sebelumnya. Bila pada tahap ini prototype masih belum diterima, maka tahapan 2 dan 3 akan diulang hingga akhirnya dapat memenuhi seluruh kebutuhan sang pemakai.

e. Implementasi sistem.

Perangkat lunak yang telah teruji dan diterima dengan baik oleh pemakai sudah siap digunakan untuk keperluan sang pemakai.

3.2.3.3 Alat Bantu Analisis dan Perancangan

1) *Actor*

Menurut Bambang Heriyanto (2004:269) Dalam pemodelan sistem dengan UML, actor adalah seseorang atau sesuatu yang berinteraksi dengan sistem. Yang dimaksud dengan berinteraksi adalah actor mengirim atau menerima pesan kea tau dari sistem atau mempertukarkan informai dengan sistem.

Secara prinsip dapat kita kenali 3 jenis actor untuk hampir semua sistem/perangkat lunak yang kita kembangkan, yaitu sebagai berikut :

1. Aktor yang pertama yaitu orang-orang yang hadir secara fisik, atau para pengguna. Mereka adalah aktor yang paling umum dan hadir di setiap sistem/perangkat lunak.
2. Aktor yang kedua yaitu sistem lain.
3. Aktor yang ke tiga yaitu waktu menjadi aktor ketika ia memicu event-event tertentu bagi sistem/perangkat lunak yang kita kembangkan.

2) *Use-case Diagrams*

Use-case Diagrams digunakan untuk mendeskripsikan apa yang seharusnya dilakukan oleh sistem.

Menurut Bambang Heriyanto (2004:267) Use Case adalah interaksi antara actor eksternal dan sistem, hasil yang dapat diamati oleh actor, berorientasi pada tujuan, dideskripsikan di diagram use-case dan teks.

Use-case Diagrams menyediakan cara medeskripsikan pandangan eksternal terhadap sistem dan interaksi-interaksinya dengan dunia luar. Dengan cara ini diagram *use-case* menggantikan diagram konteks pada pendekatan konvensional. Untuk penciptaan model *use-case* melibatkan pendefinisian sistem, pencarian aktor-aktor dan *use-case*, mendeskripsikan *use-case* dan mendefinisikan hubungan antar *use-case* dan terakhir adalah melakukan validasi model.

Use-case bertindak sebagai mekanisme terstuktur untuk diagram-diagram interaksi. Umumnya, satu diagram interaksi digambarkan untk masing-masing *use-case* diagram. Salah satu bahaya *use-case* merupakan bagian vital dari pengembangan berorientasi objek. Kita seharusnya menggunakannya setiap ingin memahami kebutuhan-kebutuhan sistem.

3) *Class Diagrams*

Class menggambarkan keadaan (atribut / properti) suatu sistem, sekaligus menawarkan layanan untuk memanipulasi keadaan tersebut (metoda / fungsi).

Class Diagram mendeskripsikan jenis-jenis objek dalam sistem dan berbagai macam hubungan statis yang terdapat diantara mereka. *Class Diagram* juga menunjukkan property dan operasi sebuah class dan batasan-batasan yang terdapat dalam hubungan-hubungan objek tersebut. UML menggunkan istilah fitur sebagai istilah umum yang meliputi property dan operasi sebuah *class*.

Class diagrams menggambarkan struktur dan deskripsi *class*, package dan objek beserta hubungan satu sama lain seperti containment, pewarisan, asosiasi, agregasi dan hubungan dinamis.

Class memiliki tiga area pokok, yaitu:

a) Nama

b) Atribut

c) Metoda

Class dapat merupakan implementasi dari sebuah interface, yaitu *class* abstrak yang hanya memiliki metoda. Interface tidak dapat langsung diinstansiasikan, tetapi harus diimplementasikan dahulu menjadi sebuah *class*. Dengan demikian interface mendukung resolusi metoda pada saat *run-time*.

Hubungan antar *Class*:

a. Asosiasi, yaitu hubungan statis antar class. Umumnya menggambarkan *class* yang memiliki atribut berupa *class* lain.
b. Agregasi, yaitu hubungan yang menyatakan bagian.
c. Pewarisan, yaitu hubungan hierarkis antar *class*. Class dapat diturunkan dari class lain dan mewarisi semua stribut dan metoda class asalnya dan menambahkan fungsionalitas baru, sehingga ia disebut anak dari *class* yang diwarisinya.
d. Hubungan dinamis, yaitu rangkaian pesan yang di-*passing* dari satu *class* kelapada *class* lain.

4) *Object Diagrams*

Object diagrams memodelkan pandangan statik terhadap sistem untuk memodelkan struktur objek.

Menurut Bambang Heriyanto (2004:291) *Object Diagram* adalah diagram instan (instance diagram) yang mendeskripsikan instan-instan kelas.

Pemodelan struktur objek melibatkan *snapshot* dari objek-objek sistem pada suatu waktu. *Object diagrams* merepresentasikan satu frame statik papn cerita dinamis dari diagram interaksi.

Kegunaan object diagram adalah mendeskripsikan bagaimana kumpulan objek tertentu saling berhubungan. *Object diagrams* adalah diagram instan yang mendeskripsikan instan-instan kelas. Diagram instan berguna untuk

dokumentasi skenario, serta kasus pengujian dan mendiskusikan contoh-contoh instanisasi *class diagrams*.

5) *Component Diagrams*

Menurut Bambang Heriyanto (2004:293) Komponen adalah bagian fisik dan dapat diganti sistem yang memenuhi dan menyediakan realisasi sekumpulan antar muka.

Component Diagrams menunjukan organisasi dan kebergantungan di antara sekumpulan komponen. Diagram ini memodelkan pandangan implementasi fisik dari sistem. *Component diagrams* berisi:

a. Komponen
b. Antarmuka
c. *Dependency*, generalisasi, asosiasi dan realisasi
d. Paket, untuk mengelompokkan elemen-elemen model menjadi potongan-potongan besar.

6) *Deployment Diagrams*

Deployment Diagrams digunakan untuk memodelkan aspek fisik dari sistem berorientasi objek, yaitu memodelkan konfigurasi node-node pengolahan waktu jalan dan komponen-komponen yang tinggal di node-node itu.

Deployment Diagram menunjukkan susunan fisik sebuah sistem, menunjukkan bagian perangkat lunak mana yang berjalan pada perangkat keras mana. Deployment diagram sangatlah sederhana.

Deploy Diagrams bisa juga berisi komponen-komponen, masing-masing komponen itu berada di suatu node. Juga dapat berisi paket atau subsistem yang digunakan mengelompokkan elemen-elemen di model menjadi potongan-potongan.

Penggunaan model *deployment* yaitu:

a. Memodelkan *embedded system*

b. Memodelkan sistem *client/server*

c. Memodelkan sistem tersebar penuh

7) *Sequence Diagrams*

Sequence diagram terdiri antar dimensi vertikal (waktu) dan dimensi horizontal (objek-objek yang terkait).

Sebuah *Sequence diagram*, secara khusus, menjabarkan behavior sebuah scenario tunggal. Diagram tersebut menunjukkan sebuah objek contoh dan pesan-pesan yang melewati objek-objek ini di dalam *use case*.

*Sequence diagram*s biasa digunakan untuk menggambarkan skenario atau rangkaian langkah-langkah yang dilakukan sebagai *respons* dari sebuah *event* untuk menghasilkan *output* tertentu.

8) *Collaboration Diagrams*

Collaboration diagrams mendefinisikan peran-peran yang dimainkan letika satu tugas dilakukan. Peran-peran dimainkan oleh instan-instan yang berinteraksi.

Collaboration diagrams menyatakan komunikasi diantara objek-objek yang menunjukan pesan-pesan yang ada, urutan pesan dan hubungan antar objek-objek.

Collaboration diagrams digunakan untuk:

a. Pandangan dalam arti perilaku sistem, berfokus pada link-link di antara objek-objek
b. Ilustrasi dari suatu diagram *use-case*
c. Menyatakan objek-objek yang diperlukan untuk merealisasikan suatu layanan
d. Memeriksa jalur-jalur pengaksesan.

Collaboration diagrams menekankan pada organisasi objek yang berpartisipasi di interaksi. *Sequence diagrams* dan *collaboration diagrams* sama-sama menunjukkan interaksi (aspek dinamis). *Sequence diagrams* fokus pada waktu sedangkan *collaboration diagrams* fokus pada ruang. Sebagaimana *sequence diagrams, collaboration diagrams* juga dapat digunakan untuk mengilustrasikan eksekusi satu operasi, *use-case* atau skenario interaksi di sistem.

9) *Activity Diagrams*

Menurut Bambang Heriyanto (2004:325) *Activity diagrams* adalah diagram flowchart yang diperluas yang menunjukkan aliran kendali satu aktivitas ke aktivitas lain. Kita menggunakan diagram ini untuk memodelkan aspek dinamis sistem. Aktivitas adalah eksekusi nonatomik yang berlangsung di *state machine*. Diagram aktivitas mendeskripsikan aksi-aksi dan hasilnya. Diagram aktivitas berupa operasi-operasi dan aktivitas-aktivitas di *use-case*.

Activity diagram dapat digunakan untuk:

a. Pandangan dalam yang dilakukan di operasi.
b. Pandangan dalam bagaimana objek-objek bekerja.
c. Pandangan dalam di aksi-aksi dan pengaruhnya pada objek-objek.
d. Pandangan dalam dari suatu *use-case.*
e. Logik dari proses bisnis.

3.2.4 Pengujian Software

Untuk Pengujian Software, penulis akan menggunakan Metode Black Box karena merupakan metode yang sangat efektif untuk pengujian perangkat lunak yang dihasilkan.

Pengujian Black Box berkaitan dengan pengujian yang dilakukan pada interface perangkat lunak, meskipun didesain untuk mengungkap kesalahan, pengujian Black Box digunakan untuk memperlihatkan bahwa fungsi-fungsi perangkat lunak adalah

operasional, bahwa input diterima dengan baik dan output dihasilkan dengan tepat, dan integritas informasi eksternal (seperti file data) dipelihara.

Menurut Bambang Heriyanto (2004:569) Pengujian adalah proses pemeriksaan atau evaluasi sistem atau komponen sistem secara manual atau otomatis untuk memverifikasi apakah sistem memenuhi kebutuhan-kebutuhan yang dispesifikasikan atau mengindentifikasi perbedaan-perbedaan antara hasil yang diharapkan dengan yang terjadi.

Pengujian black box adalah pengujian aspek *fundamental* sistem tanpa memperhatikan struktur logika internal perangkat lunak. Metode ini digunakan untuk mengetahui apakah perangkat lunak berfungsi dengan benar.

Pada pengujian *black-box*, kasus-kasus pengujian berdasarkan pada spesifikasi sistem. Rencana pengujian dapat dimulai sedini mungkin di proses pengembangan perangkat lunak.

Pada pengujian *black box,* mencoba beragam masukan dan memeriksa keluaran yang dihasilkan. Teknik pengujian *black box* juga dapat digunakan untuk pengujian berbasis skenario, dimana isi dalam sistem mungkin tidak tersedia untuk diinspeksi tapi masukan dan keluaran yang didefinisikan dengan *use case* dan informasi analisis yang lain.

Adapun faktor-faktor pengujian *black-box* adalah :

1. *File integrity*

   Adalah data yang dimasukkan melalui aplikasi akan tidak bias diubah. Prosedur yang akan memastikan bahwa file yang digunakan benar dan data dalam file tersebut akan disimpan sekuensial dan benar.

2. *Service levels*

   Menekankan bahwa hasil yang diinginkan didapat dalam waktu yang diinginkan oleh *user*. Untuk mencapai keinginan tersebut, harus dilakukan penyesuaian antara keinginan *user* dengan sumber daya yang ada.

3. *Ease of use*

   Menekankan perluasan usaha yang diminta untuk belajar, mengoprasikan dan menyiapakan inputan, dan menginterpretasikan output dari sistem. Faktor ini tersangkut dengan *usability* system terhadap interaksi antara manusia dan system.

4. *Authorization*

   Menjamin data diproses sesuai dengan ketentuan manajemen. *Authorization* menyangkut proses transaksi secara umum dan khusus.

Fokus Pengujian *Black box testing* yaitu sebagai berikut :

a) Menguji fungsi-fungsi khusus dari aplikasi.
b) *Test input* dan *output* untuk fungsi yang ada tanpa memperhatikan prosesnya.

Beberapa jenis kesalahan yang dapat di identifikasi :

1) Fungsi tidak benar atau hilang,
2) Kesalahan antar muka,

3) Kesalahan pada struktur data (pengaksesan basis data),

4) Kesalahan inisialisasi dan akhir program, dan

5) Kesalahan performasi.