BAB II
LANDASAN TEORI

2.1 **Tinjauan Pustaka**

Beberapa tinjauan pustaka terkait penelitian yaitu, Sistem komputerisasi pencataatan stok laptop di internasional *computer* penelitian ini dilakukan untuk membuat suatu sistem komputersasi pencatatan stok laptop di internasional computer berbasis web, sistem yag di buat ini memiliki tiga level login, yaitu admin, petugas dan pemilik, yang ketiga level tersebut memiliki fitur yang berbeda beda admin memiliki fitur mengelola data user, petugas meiliki fitur mengelola data barang dan data supplier, sedangkan untuk pemimpin memiliki fitur melihat data laporan stok dan laporan transaksi. (Dananjaya, 2017)

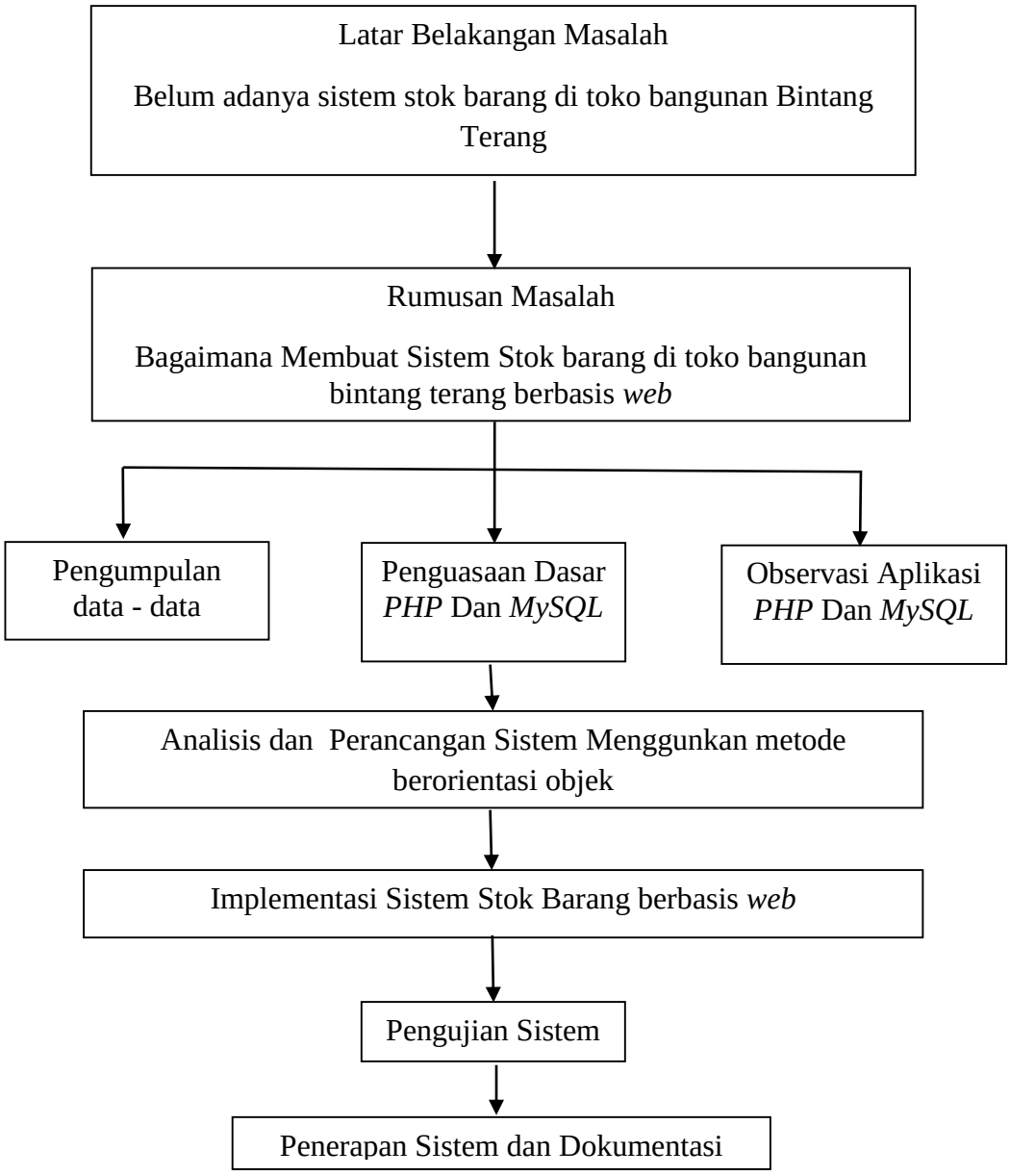
Septiyanto (2019) Membuat sistem inventaris berbasis web guna menajemen barang di Universitas Sahid Surakarta, di dalam desain sistem inventarsi yang di buat terdapat penyimpanan data barang, barcode barang, transaksi peminjaman, pengembalian, pengandaan, penempatan, mutasi barang dan menu melihat seluruh laporan, dalam sistem yang di buat meiliki tiga user yaitu, admin bagian umum sebagai pengelola penuh sistem, unit sebagai peminjam dan petugas untuk melihat laporan data inventasi barang.

Setiwati (2010) Membuat aplikasi stok barang berbasis web di gudang speart part pada PT. Arwana Citramulia Tbk Tanggerang, dalam sistem yang di buat di PT. Arwana Citramulia Tbk ini menfasilitasi transaksi transaksi diantaranya *memo request, purchase request, purchase order,* bukti penerimaan barang dan bukti pengambilan barang yang dapat memepermudah jalannya stok barang di gudang selain itu sistem yang sudah di buat juga di lengkapi tentang info info di PT. Arwana Citramulia Tbk.

2.2 Kerangka Berfikir

Berikut ini adalah tahapan kerangka pemikiran pada proses pembuatan sebuah sistem stok barang di toko bangunan bintang terang yang akan memudahkan dalam

perancangan sistem sampai sampai denagan implemtasi sistem yang dibuat dan dijalankan oleh penulis kerangka pemikiran di jelaskan pada Gambar 2.1



Gambar 2.1 Diagram Kerangka Pemikiran

Uraian dari kerangka berfikir sebagai berikut :

1. Latar Belakang Masalah

Latar Belakang masalah pada tugas akhir ini adalah belum adanya sistem stok barang di toko bangunan Bintang Terang yang menyebabkan kurangnya efisensi

waktu dalam pendataan stok barangf dan pencarian data stok barang yang di butuhkan .

2. Rumusan Masalah

Rumusan masalah pada tugas akhir ini adalah bagaiman cara membuat sistem stok barang di toko bangunan Bintang Terang berbasis *web* yang dapat memberikan informasi tentang stok barang yang terdapat di toko bangunan Bintang Terang.

3. Pengumpulan Data

Pengumpulan data dalam penelitian ini ada dua yaitu pengumpulan data tertulis dan pengumpulan data tidak tertulis. Penelitian ini mengumpulkan semua data yang dibutuhkan dengan melakukan *interview* dengan pemilik toko untuk mendapatkan informasi tentang apa saja yang di butuhkan untuk membangun sistem stok barang.

4. Penguasaan Dasar

Penguasaan dasar dalam penelitian ini adalah penguasaan yang telah diketahui atau dikuasai penulis mengenai *PHP* dan *MySQL*.

5. Observasi Aplikasi

Observasi alat dalam penelitian ini adalah penguasaan terhadap *software* - *software* yang akan digunakan untuk membuat sistem stok barang di toko bangunan Bintang Terang .

6. Analisis dan Perancangan Sistem

Analisis dan perancangan sistem dalam penilitian ini menggunakan metode berorientasi objek yaitu menggunkan *UML* dengan menggunkan *use case digram, class diagram, sequence diagram, activity diagram, component diagram, dan depoloment diagram* agar mempermudah dalam mengalisis dan merancang sebuah sistem yang baru.

7. Implementasi Sistem

Setelah dilakukan analisis dan perancangan kemudian dilakukan sebuah implementasi sistem dalam penelitian ini adalah poroses pelaksanaan pembuatan sebuah sistem stok barang berbasis web setelah penilitian yang dilakukan.

8. Pengujian Sistem

Pengujian sistem dalam penelitian ini menggunkan *black box* yaitu menguji perangkat lunak dari segi spesifikasi funsionla tanpa tahap menguji desain dank ode program, dilakukan untuk mengetahui apabila dalam sistem masih ada kesalahan atau kekurangan pada sistem yang dibuat.

9. Penerapan Sistem dan Dokumentasi

Penerapan sistem dan dokumentasi dalam penelitian ini adalah proses menerapkan aplikasi di toko bangunan Bintang Terang, proses pengambilan dokumentasi setelah sistem selesai dibuat dari keseluruhan kegiatan Tugas Akhir.

2.3 Dasar Teori

2.3.1 Sistem

Sistem adalah sebuah tatanan (keterpaduan), sekelompok undur atau elemen yang berhubungan satu dengan yang lain untuk mencapai suatu tujuan (Rahmawati, 2017).

2.3.2 Sistem Informasi

Sistem infomasi dapat didefinisikan sebagai serangkain komponen yang saling berhubungan yang mengumpulkan , memproses, menyimpan, dan mendistribusikan informasi yang mendukung pengambilan keputusan dan pengawasan di dalam sebuah organisasi(Rahmawati, 2017).

2.3.3 Stok atau Persediaan

Pengertian dari stok atau persediaan adalah sebagai suatu aktiva yang meliputi barang-barang milik perusahaan dengan maksud untuk di jual dalam periode usaha yang normal(Rahmawati, 2017).

2.3.4 *PHP (Personal Home Page)*

PHP merupakan bahasa pemrograman pelengkap *HTML* (*Hypertext Markup Languange)* yang memungkinkan aplikasi *web* dinamis untuk pengolahan data, pemrosesan data dari *user* via *form*, membuat buku tamu, toko *online*, dan lain

sebagainya, dengan mudah *PHP* dapat melakukan koneksi ke *database* karena *PHP* memang dilengkapi fitur yang memungkinkan koneksi ke *PHP* dilakukan dengan mudah, tanpa harus melakukan pemrograman yang memusingkan. *PHP* juga merupakan bahasa pemrograman berbasis *server side* yang dapat melakukan parsing *script PHP* menjadi *script web* sehingga dari sisi
client menghasilkan suatu tampilan yang menarik.

Jadi dapat disimpulkan bahwa pengertian *Personal Home Page* (*PHP*) adalah bahasa pemrograman pelengkap *HTML* berbasis *server side* yang memungkinkan aplikasi *web* dinamis, dapat melakukan koneksi ke *database* dan menghasilkan suatu tampilan yang menarik (Rahmawati, 2017).

2.3.5 *HTML (Hypertext Markup Language)*

HTML adalah sebuah bahasa markah yang digunakan untuk membuat sebuah halaman *web*, menampilkan berbagai informasi di dalam sebuah penjelajah *web* internet dan pemformatan hiperteks sederhana yang ditulis dalam berkas format ASCII agar dapat menghasilkan tampilan wujud yang terintegrasi.
HTML adalah bahasa *markup* untuk menstrukturkan dan menampilkan isi dari *World Wide Web*, sebuah teknologi inti dari internet.

Jadi dapat disimpulkan bahwa pengertian *Hypertext Markup Languange* (*HTML*) adalah sebuah bahasa *markup* untuk membuat sebuah halaman *web* yang menampilkan berbagai informasi untuk menstrukturkan dan menampilkan isi dari *World Wide Web* pada *browser* yang ditulis dalam berkas format *ASCII* (Rahmawati, 2017).

2.3.6 *MYSQL*

MYSQL adalah salah satu jenis database server yang sangat terkenal dan banyak digunakan untuk membagun aplikasi web yang menggunakan database sebagai sumber dan pengolahan datanya. *MYSQL* merupakan basis data yang berkembang dari bahasa *SQL (Stucture query language)*. *MYSQL* dapat dikatakan sebagai relational database management system (RDBMS), yaitu hubungan antar

table yang berisi data data pada suatu database. Dengan demikian dapat mempercepat pencarian suatu data. *SQL* merupakan bahasa tersetruktur yang digunakan untuk interaksi antar script program dengan database server dalam hal pengolahan data.(Arief, 2011)

2.3.7 Databases

Database merupakan sebuah tempat untuk menyimpan data yang jenisnya beraneka ragam. Keuntungan menyimpan data di *database* adalah kemudahannya dalam menyimpan dan menampilkan data dalam bentuk table (Winarno, dkk, 2014).

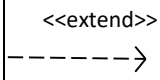
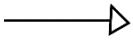
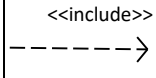
2.3.8 UML

Pada perkembangan teknologi perangkat lunak diperlukan adanya bahasa yang digunakan untuk memodelkan perangkat lunak yang akan dibuat dan perlu adanya standarisasi agar orang diberbagai Negara dapat mengerti pemodelan perangkat lunak. Pada perkembangan teknik pemrograman berorientasi objek, munculah sebuah standarisasi bahasa pemodelan untuk pembangunan perangkat lunak yang dibangun dengan menggunkan teknik pemrograman berorientasi objek, yaitu *Unified Modelling Language* (Rosa dan Shalahudin, 2014).

2.3.8.1 Use Case Diagram

Use case atau *diagram use case* merupakan pemodelan untuk melakukan sistem informasi yang akan dibuat. *Diagram use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. (Rosa dan Shalahudin, 2014). Simbol-Simbol yang ada pada diagram *use case* dapat dilihat pada Tabel 2.1.

Tabel 2.1. Simbol – Simbol *Use Case Diagram*

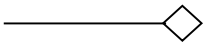
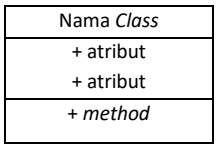
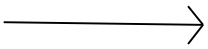
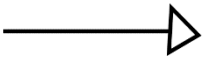
NO.	SIMBOL	NAMA	KETERANGAN
1.		*Actor*	Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat diluar sistem informasi yang akan dibuat itu sendiri.
2.		*Association*	Komunikasi antara aktor dan *use case* yang berpartisipasi pada *use case* memiliki interaksi dengan aktor.
3.		*Use Case*	Fungsionalitas yang disediakan sistem sebagai unit yang saling bertukar pesan antar unit atau aktor.
4.		*Extend*	Relasi *use case* tambahan ke sebuah *use case* yang ditambahkan dapat berdiri sendiri walau tanpa *use case* tambahan itu.
5.		*Generalization*	Hubungan generalisasi dan spesifikasi (umum-khusus) antara dua buah *use case* dimana fungsi yang satu adalah fungsi yang lebih umum dari lainnya.
6.		*Include*	Relasi *use case* tambahan ke sebuah *use case* dimana *use case* yang ditambahkan memerlukan *use case* ini untuk menjalankan fungsinya atau sebagai syarat dijalankan *use case* ini.

2.3.8.2 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 operasi (Rosa dan Shalahudin, 2014). Simbol-simbol yang ada pada *class diagram* dapat dilihat pada Tabel 2.2.

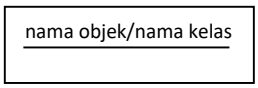
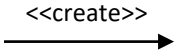
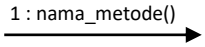
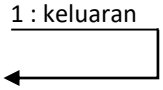
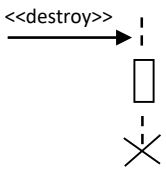
Tabel 2.2. Simbol – Simbol *Class Diagram*

NO.	SIMBOL	NAMA	KETERANGAN
1.		*Aggregation*	Relasi antar kelas dengan makna semua bagian *(whole-part)*
2.		*Class*	Kelas pada struktur sistem.
3.		*Association*	Relasi antar kelas dengan makna umum, asosiasi biasanya juga disertai dengan *multiplicity.*
4.		*Dependency*	Relasi antar kelas dengan makna kebergantungan antar kelas.
5.		*Directed Association*	Relasi antar kelas dengan makna kelas yang satu digunakan oleh kelas lain. Asosiasi biasanya juga disertai dengan *multiplicity.*
6.		*Generalization*	Relasi antar kelas makna generalisasi-spesialisasi (umum khusus).
7.		*Interface*	Sama dengan konsep *interface* dalam pemrograman berorientasi objek.

2.3.8.3 Sequence Diagram

Diagram sekuen menggabarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan *message* yang dikirimkan dan diterima antar objek (Rosa dan Shalahudin, 2014). Simbol-simbol yang ada pada *sequence* diagram dapat dilihat pada Tabel 2.3

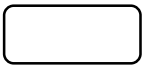
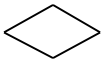
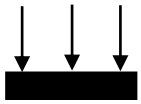
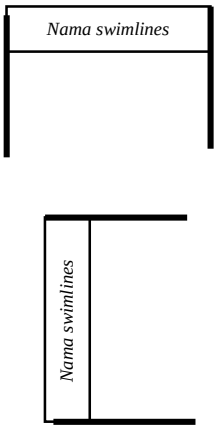
Tabel 2.3. Simbol – Simbol *Sequence Diagram*

NO.	SIMBOL	NAMA	KETERANGAN
1.		Aktor	Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat di luar sistem informasi yang akan dibuat itu sendiri, jadi walaupun symbol dari actor adalah orang, belum tentu merupakan orang biasanya dinyatakan menggunkan kata benda di awal frase nama aktor.
2.		*Lifeline*	Menyatakan kehidupan suatu objek.
3.		Objek	Menyatakan objek yang berinteraksi pesan.
4.		Waktu aktif	Menyatakan objek dalam keadaan aktif dan berinteraksi.
5.		Pesan tipe *create*	Menyatakan suatu objek membuat objek yang lain, arah panah mengarah pada objek yang dibuat
6.		Pesan tipe *call*	Menyatakan bahwa suatu objek mengirimkan data / masukan / informasi ke objek lainya, arahg panah mengarah pada objek yang dikirim.
7.		Pesan tipe *send*	Menyatakan bahwa suatu objek mengirimkan data / masukan / informasi ke objek lainnya, arah panah mengarah pada objek yang dikirimi.
8.		Pesan tipe *return*	Menyatakan bahwa suatu objek telah menjalankan suatu operasi atau metode menghasilkan suatu kembalian ke objek tertentu, arah panah ke objek kembalian.
9.		Pesan tipe *destroy*	Menyatakan suatu objek mengakhiri hidup objek lain, arah panah mengarah pada objek yang diakhiri, sebaiknya jika ada *create* maka ada *destroy.*

2.3.8.4 Activity Diagram

Diagram aktivitas atau *Activity diagram* menggambarkan *workflow* atau aktivitas dari sebuah sistem atau proses bisnis atau menu yang ada pada perangkat lunak. Yang perlu diperhatikan disini diagram aktivitas menggambarkan aktivitas sistem bukan apa yang dilakukan aktor (Rosa dan Shalahudin, 2014). Simbol simbol yang ada pada *activity* diagram dapat dilihat Pada Tabel 2.4.

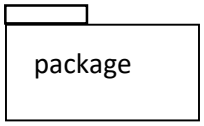
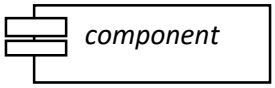
Tabel 2.4. Simbol – Simbol *Activity Diagram*

NO	GAMBAR	NAMA	KETERANGAN
1.		Status Awal	Status awal aktivitas sistem.
2.		Status Akhir	Status akhir dilakukan sebuah sistem.
3.		Aktivitas	Aktivitas yang dilakukan sistem, biasanya diawali dengan kata kerja.
4.		Decision	Asosiasi percabangan dimana jika ada pilihan aktivitas lebih dari satu.
5.		*Join*	Asosiasi penggabungan lebih dari satu aktivitas digabungkan jadi satu.
6.		*Swimlanes*	Memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi.

2.3.8.5 ***Component Diagram***

Diagram komponen *atau component* diagram dibuat untuk menunjukkan organisasi dan ketergantungan diantara kumpulan komponen dalam sebuah sistem (Rosa dan Shalahudin, 2014). Simbol-simbol yang ada pada *component* diagram dapat dilihat pada Tabel 2.5

Tabel 2.5. Simbol – Simbol *Component Diagram*

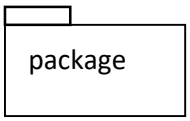
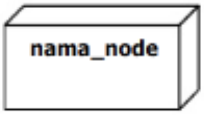
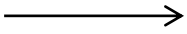
NO	GAMBAR	NAMA	KETERANGAN
1.		*Package*	*Package* merupakan sebuah bungkusan dari satu atau lebih komponen
2.		*Component*	Komponen sistem
3.		*Dependency*	Kebergantungan antar komponen, arah panah mengarah pada komponen yang dipakai
4.		*Interface*	Sama dengan konsep *interface* pada pemrograman berorientasi objek, yaitu sebagai antarmuka komponen agar tidak mengakses langsung komponen
5.		*Link*	Relasi antar komponen

2.3.8.6 Deployment Diagram

Deployment diagram menunjukan konfigurasi komponen dalam proses eksekusi aplikasi. Diagram *deployment* juga dapat digunakan untuk memodelkan sistem tambahan, sistem *client*, sistem terdistribusi murni, rekayasa ulang aplikasi

(Rosa dan Shalahudin, 2014). Simbol-simbol yang ada pada *deployment* diagram dapat dilihat pada Tabel 2.6.

Tabel 2.6. Simbol – Simbol *Deployment Diagram*

NO	SIMBOL	NAMA	KETERANGAN
1.		*Package*	*Package* merupakan sebuah bungkusan dari satu atau lebih *node*.
2.		*Node*	Biasanya mengacu pada perangkat keras (*hardware*), perangkat lunak yang tidak dibuat sendiri (*software*), jika didalam *node* disertakan komponen untuk mengkonsistenkan rancangan maka komponen yang diikutsertakan harus sesuai dengan komponen yang telah didefinisikan sebelum pada diagram komponen.
3.		*Dependency*	Kebergantungan antar *node*, arah panah mengarah pada *node* yang dipakai.
4.		*Link*	Relasi antar *node*.

2.4 Black Box Testing

Pengertian *blackbox testing* yaitu menguji perangkat lunak dari segi spesifikasi fungsional tanpa menguji desain dan kode program. Pengujian ini

dimaksudkan untuk mengetahui apakah fungsi-fungsi, masukan, dan keluaran dari penrangkat lunak sesuai dengan spesifikasi yang dibutuhkan.

Pengujian kotak hitam dilakukan dengan membuat kasus uji yang bersifat mencoba semua fungsi dengan memakai perangkat lunak apakah sesuai dengan spesifikasi yang dibutuhkan. Kasus uji coba yang dibuat untuk melakukan pengujian kotak hitam harus dibuat dengan kasus benar dan kasus salah, misalkan untuk kasus proses *login* maka kasus uji yang dibuat adalah :

a. Jika *user* memasukkan nama pemakai (*username)* dan kata sandi (*password)* yang benar.
b. Jika *user* memasukkan nama pemakai *(username)* dan kata sandi *(password)* yang salah, misal nama pemakai benar kata sandi salah atau sebaliknya, atau keduanya salah (Winarno, dkk., 2014).

Pengujian *black-box* berfokus pada persyaratan fungsional perangkat lunak. Dengan demikian pengujian *black-box* memungkinkan perekayasa perangkat lunak mendapatkan serangkaian kondisi *input* yang sepenuhnya menggunakan semua persyaratan fungsional untuk suatu program. Pengujian black-box bukan merupakan alternatif dari teknik *white-box,* tetapi merupakan pendekatan komplementer yang kemungkinan besar mampu mengungkap kelas kesalahan daripada metode *white-box.*

Pengujian *black-box* berusaha menemukan kesalahan dalam kategori sebagai berikut: (1) fungsi-fungsi yang tidak benar atau hilang, (2) kesalahan *interface,* (3) kesalahan dalam struktur data atau akses *database external*, (4) kesalahan kinerja, (5) inisialisasi dan kesalahan terminasi.

Pengujian *black-box* memperhatikan struktur kontrol, maka perhatian berfokus pada *domain* informasi. Pengujian di desain untuk menjawab pertanyaan-pertanyaan berikut :

1. Bagaimana validitas fungsional diuji ?
2. Kelas inut apa yang akan membuat *test case* menjadi baik ?
3. Apakah sistem sangat sensitif terhada harga input tertentu ?
4. Bagaimana batasan dari suatu data diisolasi ?

5. Kecepatan data apa dan volume data apa yang dapat ditolelir oleh sistem ?

   Dengan mengaplikasikan teknik *black-box*, maka serangkaian *test case* akan memenuhi kriteria berikut ini : (1) *test case* yang mengurangi, dengan harga lebih dari satu, jumlah *test case* tambahan yang harus didesain untuk mencapai pengujian yang dapat dipertanggungjawabkan, dan (2) *test case* yang memberi tahu mengenai kehadiran atau ketidakhadiran kelas kesalahan, daripada memberitahu kesalahan yang berhubungan hanya dengan pengujian sesifik yang ada.