# BAB III

# OBJEK DAN METODE PENELITIAN

## 3.1 Objek Penelitian

CV REDSDIPO INDONESIA adalah Rumah Makan yang menjual berbagai macam makanan seafood diantaranya ikan, cumi, udang, selain itu ada juga menu ayam goreng / bakar dan menu minuman berbagai macam jus buah-buahan sebagai penyegar. Demi mengembangkan bisnisnya, Redsdipo membuka pelayanan pemesanan / catering yang bertujuan meningkatkan profit dan nilai jual bisnisnya. Maka dari itu mobilitas CV REDSDIPO INDONESIA sangat tinggi, mobilitas yang sangat tinggi itu tidak didukung oleh sistem penggajian dan sistem laporan hasil penjualan yang baik.

### 3.1.1 Sejarah CV REDSDIPO INDONESIA

Awal berdirinya Redsdipo merupakan cerita yang unik. Berawal dari Aprianto Wibowo mahasiswa UNPAR (Universitas Parahyangan) yang mendapati masalah terhadap perkuliahannya sehingga terpaksa harus berhenti sementara / cuti selama 1tahun. Dikarenakkan perhentian yang cukup lama, salah seorang pendiri Redsdipo tersebut terlintas ide untuk membuat sebuah warung makan pinggir jalan didasari pada hobi memasaknya dan mengajak kerjasama teman-teman yang lain yaitu, Posma Abraham, Johan Pangaribuan, Rendi Koeswara. Dengan bermodalkan Rp 2.000.000 untuk mendirikan warung makan pinggir jalan dan resep masakan dari sang ibunda tercinta, berdirilah waroeng bawal Redsdipo tahun 2003 di Jalan Diponegoro.

Redsdipo sendiri merupakan nama yang dipilih karena memiliki filosofi tersendiri jika menurut pendiri kata Reds di pilih karena para pendiri merupakan pendukung salah satu klub bola ternama yaitu Liverpool, dan kata Dipo di pilih karena tempat awal berdirinya warung makan pinggir jalan di Jl.Diponegoro. Namun artian lain dari nama ini Reds dikarenakan seluruh tenda & seluruh peralatan berwarna merah, dan Dipo karena bertempatkan di Jl.Diponegoro.

Awal dari usaha ini tidaklah berjalan lancar, mereka mendapati kendala bahwa selama 2 minggu awal usaha tidak berjualan sama sekali. Namun hal ini tidak mematahkan semangat untuk tetap berusaha, hingga pada akhirnya waroeng bawal redsdipo ini pun mengalami kemajuan yang pesat. Keistimewaan warung makan ini pun terletak pada citarasa sambel yang disebut Sambel Tyson karena sambel ini sangat pedas. Selama tahun 2003-2007 waroeng bawal redsdipo ini terletak di jalan diponegoro, karena kemajuan yang pesat pendiri pun menyempurnakan struktur organisasinya demi kemajuan warung tersebut. Dan pada tahun 2007 lokasi pun pindah ke Jl. Dipatiukur No. 1 dengan tempat yang jauh lebih luas dan nyaman sehingga menu yang disediakan lebih beragam.

### 3.1.2 Visi Dan Misi CV REDSDIPO INDONESIA

Visi CV REDSDIPO INDONESIA

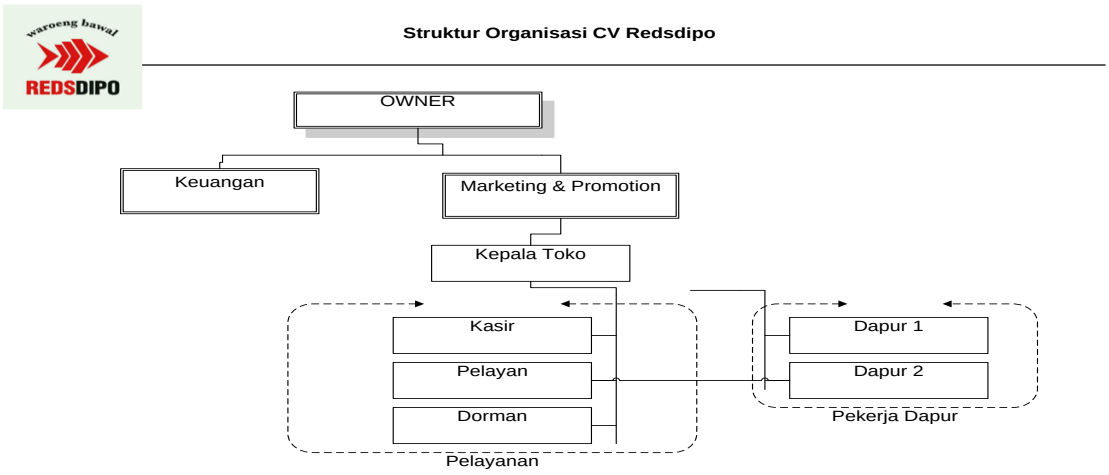
*“ Menciptakan Usaha Kuliner terbaik di Bandung dan membentuk kemandirian”*

Misi CV REDSDIPO INDONESIA :

1. Menberikan kemudahan dalam melayani kebutuhan pokok.
2. Menciptakan lapangan pekerjaan.
3. Membangun semangat enterprenuer / berwirausaha.
4. Membangun kemandirian mahasiswa dalam menghadapi tantangan global.
5. Terciptanya ekonomi bersama yang mandiri.
6. Terbentuknya usaha kuliner terbaik yang mengedepankan Pelayanan. Kenikmatan dan Keberkahan

### 3.1.3 Struktur Organisasi CV REDSDIPO INDONESIA

Dengan mengetahui struktur organisasi suatu perusahaan atau instansi pemerintah ataupun swasta maka akan dapat diperoleh gambaran dan pengetahuan tentang bagian-bagian yang ada dalam perusahaan atau instansi pemerintah ataupun swasta, serta tanggung-jawab dan wewenang dalam melaksanakan tugasnya masing-masing.



Gambar 3.1 Struktur Organisasi

Sumber : CV REDSDIPO INDONESIA

### 3.1.4 Deskripsi Tugas

**Owner :**

1. Mengawasi operasional pegawai.
2. Mengecek Jumlah stok barang yang ada di dapur.
3. Memeriksa laporan dari bagian keuangan.

**Marketing & Promotion :**

1. Operational & Promotion.
2. Mengatur operasional toko, membawahi & mengawasi kinerja pegawai bagian service (pelayan), dan bagian dapur 1&2.
3. Membuat Strategi Promotion.

**Bagian Dapur 1 :**

1. Bagian dapur 1 merupakan dapur olahan. Membuat pesanan makanan yang di masak atau diolah langsung.
2. Menerima faktur pembelian dari pelayan.
3. Mengechek persediaan pada papan stock.
4. Mengupdate persediaan di papan stock.

5. Memberitahu kepala toko persediaan habis dengan membuat surat list barang yang harus dibeli pada supplier.
6. Untuk catering menyiapkan pesanan, mengolahnya dan membungkus.
7. Apabila min order 100 porsi menyiapkan pesanan, koki yang akan diikut sertakan, dan membuat surat jalan.
8. Perwakilan (kepala dapur) membuat list & membeli bahan baku setiap harinya dengan mendatangi supplier yang terdapat di pasar.
9. Menerima faktur pembelian kemudian mengcopy & diserahkan pada kepala toko, chek barang yang di beli, apabila terdapat barang yang tidak sesuai membuat surat retur barang.
10. Membuat daftar pemakaian bahan berdasarkan papan stock & menyerahkan pada kepala toko.

**Bagian Dapur 2 :**

1. Bagian dapur 2 merupakan dapur bakaran. Membuat pesanan makanan yang di bakar.
2. Menerima faktur pembelian dari pelayan.
3. Mengechek persediaan pada papan stock.
4. Mengupdate persediaan di papan stock.
5. Memberitahu kepala toko persediaan habis dengan membuat surat list barangyang harus dibeli pada supplier.
6. Untuk catering menyiapkan pesanan, mengolahnya dan membungkus.

7. Apabila min order 100 porsi menyiapkan pesanan, koki yang akan diikut sertakan, dan membuat surat jalan.
8. Perwakilan (kepala dapur) membuat list & membeli bahan baku setiap harinya dengan mendatangi supplier yang terdapat di pasar.
9. Menerima faktur pembelian kemudian mengcopy & diserahkan pada kepala toko, chek barang yang di beli, apabila terdapat barang yang tidak sesuai membuat surat retur barang.
10. Membuat daftar pemakaian bahan berdasarkan papan stock & menyerahkan pada kepala toko.

**Bagian Keuangan :**

1. Menerima daftar hadir pegawai & rekapan omset penjualan dari kepala toko, serta menerima slip
2. Menghitung bonus & uang yang harus dikeluarkan.
3. Menyerahkan pembayaran bonus penjualan pada karyawan secara langsung beserta slip bonus.
4. Menyerahkan gaji setiap 2 minggu pada karyawan secara langsung beserta slip gaji.
5. Mengcopy slip gaji untuk arsip.
6. Membuat bukti kas keluar, copyan slip gaji dan slip bonus diserahkan pada bagian akuntansi.

## 3.2 Metode Penelitian

### 3.2.1 Desain Penelitian

Sesuai dengan tujuan yang ingin dicapai dalam penulisan laporan ini, maka metode yang digunakan adalah Metode Deskriptif dan Terapan. Metode Deskriptif adalah metode penelitian yang memberikan gambaran tentang sifat individu, keadaan, gejala suatu objek, dimana dalam penelitian ini dimaksudkan untuk memperoleh gambaran tentang sistem informasi penggajian dan juga laporan hasil penjualan pada CV Redsdipo, selain itu penelitian ini dimaksudkan untuk mengetahui kendala yang sedang dihadapi dan mencari bagaimana pemecahannya.

Sedangkan metode terapan, menguji, dan mengevaluasi kemampuan suatu teori yang diterapkan dalam pemecahan masalah-masalah kehidupan praktis.

### 3.2.2 Jenis Dan Metode Pengumpulan Data

#### 3.2.2.1 Sumber Data Primer

Teknik pengumpulan data yang digunakan oleh penulis adalah :

1. Observasi, yaitu peninjauan secara langsung ke bagian divisi penjualan dan bagian administrasi keuangan di CV REDSDIPO INDONESIA serta melihat data-data yang digunakan untuk kemudian dianalisis dan digunakan untuk pembentukan deskripsi masalah.

2. Interview (wawancara), yaitu dengan melakukan wawancara langsung dengan staf/karyawan pada bagian divisi penjualan dan bagian administrasi keuangan di CV REDSDIPO INDONESIA.

#### 3.2.2.2 Sumber Data Sekunder

Sumber data sekunder didapatkan dari dokumentasi-dokumentasi baik yang berasal dari buku ataupun dokumentasi yang berasal dari perusahaan terkait yaitu CV REDSDIPO INDONESIA.

Dokumen-dokumen yang didapat yaitu Laporan Hasil Penjualan, Slip Gaji, Catatan Penjualan per/hari, Biaya Potongan Penjualan, dan Tanda Terima.

### 3.2.3 Metode Pendekatan Dan Pengembangan Sistem

#### 3.2.3.1 Metode Pendekatan Sistem

Metode pendekatan sistem yang digunakan penulis adalah dengan menggunakan Pemodelan Berorientasi Objek yang ditandai dengan adanya diagram *Usecase Diagram*, *Activity Diagram*, *Sequential Diagram*, *Collaboration Diagram*, *Class Diagram*, *Component Diagram* dan *Deployment Diagram.*

### 3.2.3.2 Metode Pengembangan sistem *(Prototype)*

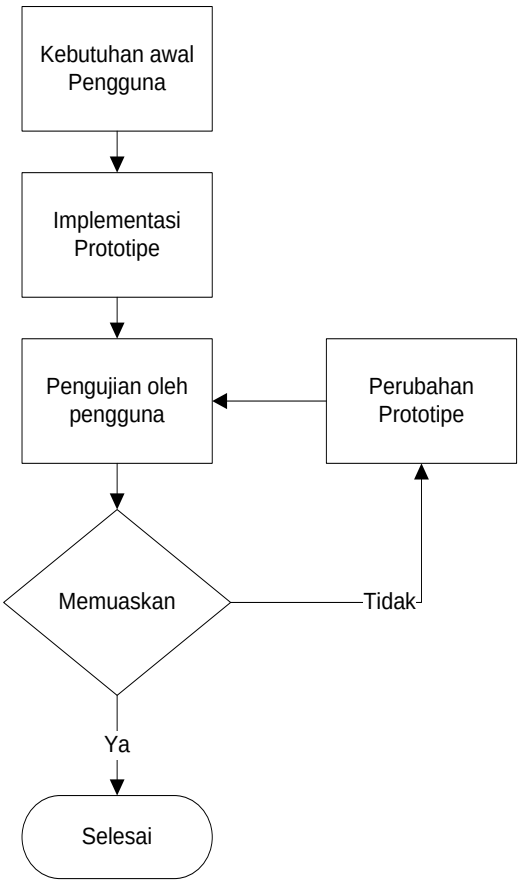
Metode *prototype* sebagai suatu paradigma baru dalam pengembangan sistem informasi manajemen, tidak hanya sekedar suatu evolusi dari metode pengembangan sistem informasi, tetapi sekaligus merupakan revolusi dalam pengembangan sistem informasi manajemen. Metode ini dikatakan revolusi karena merubah proses pengembangan sistem informasi yang lama (SDLC).

Menurut literatur, yang dimaksud dengan prototipe (*prototype*) adalah ”model pertama”, yang sering digunakan oleh perusahaan industri yang memproduksi barang secara masa. Tetapi dalam kaitannya dengan sistem informasi definisi kedua dari Webster yang menyebutkan bahwa ”*prototype is an individual that exhibits the essential peatures of later type”*, yang bila diaplikasikan dalam pengembangan sistem informasi manajemen dapat berarti bahwa *Prototype* tersebut adalah sistem informasi yang menggambarkan hal-hal penting dari sistem informasi yang akan datang. *Prototype* sistem informasi bukanlah merupakan sesuatu yang lengkap, tetapi sesuatu yang harus dimodifikasi kembali, dikembangkan, ditambahkan atau digabungkan dengan sistem informasi yang lain bila perlu.

Beberapa hal pengembangan *software*, berbeda dengan produk-produk manufaktur, setiap tahap atau fase pengembangan sistem informasi merupakan bagian yang tidak terpisahkan dari seluruh proses yang harus

dilakukan. Proses ini umumnya hanya untuk satu produk dan karakteristik dari produk tersebut tidak dapat ditentukan secara pasti seperti produk manufaktur, sehingga penggunaan ”model pertama” bagi pengembangan *software* tidaklah tepat. Istilah *prototyping* dalam hubungannya dengan pengembangan *software* sistem informasi manajemen lebih merupakan suatu proses bukan *prototype* sebagai suatu produk.

Prototipe paradigma dimulai dengan pengumpulan kebutuhan. Pengembang dan pelanggan bertemu dan mendefinisikan objektif keseluruhan dari perangkat lunak, mengidentifikasi segala kebutuhan yang diketahui, dan area garis besar dimana definisi lebih jauh merupakan keharusan kemudian dilakukan “perancangan kilat”. Perancangan kilat berfokus pada penyajian dari aspek perangkat lunak tersebut yang akan Nampak bagi pelanggan/pemakai (contohnya pendekatan input dan format *output*). Perancangan kilat membawa kepada konstruksi sebuah *prototype*. *Prototype* tersebut dievaluasi oleh pelanggan/pemakai dan dipakai untuk menyaring kebutuhan pengembangan perangkat lunak. Iterasi terjadi pada saat *prototype* disetel untuk memenuhi kebutuhan pelanggan dan pada saat yang sama memungkinkan pengembang untuk secara lebih baik memahami apa yang harus dilakukannya. Secara ideal *prototype* berfungsi sebagai sebuah mekanisme untuk mengidentifikasi kebutuhan perangkat lunak.




Gambar 3.2 Metode Prototype

Gambar 3.2. Metode Pengembangan Prototype

Sumber : Abdul Kadir : "Konsep & Tuntunan Praktis Basis Data" Th 2003

hal : 98

Karakteristik metode *prototyping* :

Ada empat langkah yang menjadi karakteristik metode *prototyping* yaitu :

- Pemilahan fungsi

Mengacu pada pemilahan fungsi yang harus ditampilkan oleh *prototyping*. Pemilahan harus selalu dilakukan berdasarkan pada tugas-tugas yang relevan yang sesuai dengan contoh kasus yang akan diperagakan.

- Penyusunan Sistem Informasi Bertujuan untuk memenuhi permintaan akan tersedianya *prototype*.
- Evaluasi penggunaan selanjutnya.

Jenis-jenis *prototyping* meliputi :

- *Feasibility prototyping* – Digunakan untuk menguji kelayakan dari teknologi yang akan digunakan untuk sistem informasi yang akan disusun.
- *Requirement prototyping* – Digunakan untuk mengetahui kebutuhan aktivitas bisnis *user*.
- Desain *Prototyping* - Digunakan untuk mendorong perancangan sistem informasi yang akan digunakan.
- *Implementation prototyping* – Merupakan lanjutan dari rancangan prototype, *prototype* ini langsung disusun sebagai suatu sistem informasi yang akan digunakan.

Sumber : Abdul Kadir : "Konsep & Tuntunan Praktis Basis Data" Th 2003

hal : 98

Beberapa kelebihan dari model Prototype :

1. Komunikasi antara analis sistem dan pemakai membaik.
2. Analis dapat bekerja lebih baik dalam menentukan kebutuhan pemakai.
3. Pemakai berperan lebih aktif dalam pengembangan sistem.
4. Spesialis informasi dan pemakai menghabiskan lebih sedikit waktu usaha dalam mengembangkan sistem.
5. Penerapan menjadi lebih mudah.

Beberapa kelemahan dari model Prototype :

1. Prototype hanya bisa berhasil jika pemakai bersungguh – sungguh dalam menyediakan waktu dan pikiran untuk mengerjakan prototype.
2. Kemungkinan dokumentasi terabaikan karena pengembangan lebih berkonsentrasi pada pengujian dan pembuatan prototype.
3. Mengingat target waktu yang pendek, ada kemungkinan sistem yang dibuat tidak lengkap atau bahkan sistem kurang teruji.
4. Jika terlalu banyak proses pengulangan dalam pembuatan prototype, ada kemungkinan pemakai menjadi jenuh dan memberikan reaksi yang negatif.
5. Apabila tidak terkelola dengan baik, prototype menjadi tidak pernah berakhir. Hal ini disebabkan permintaan terhadap perubahan terlalu mudah untuk dipenuhi.

#### 3.2.3.3 Alat Bantu Analisis dan Perancangan

Use case adalah rangkaian/uraian sekelompok yang saling terkait dan membentuk sistem secara teratur yang dilakukan atau diawasi oleh sebuah aktor.

Use case digunakan untuk membentuk tingkah-laku benda / things dalam sebuah model serta di Realisasikan oleh sebuah collaboration. Umumnya use case digambarkan dengan sebuah elips dengan garis yang solid, biasanya mengandung nama. Use case menggambarkan proses system (kebutuhan system dari sudut pandang user). Secara umum use case adalah:

- Pola perilaku system
- Urutan transaksi yang berhubungan yang dilakukan oleh satu actor

Use case diagram terdiri dari

- Use case
- Actors
- Relationship

Menurut Munawar, *Pemodelan Visual Dengan UML, 2005, Use Case* adalah deskripsi fungsi dari sebuah sistem dari perspektif pengguna. *Use Case* bekerja dengan cara mendeskripsikan tipikal interaksi antara pengguna (*User*) sebuah sistem dengan sistemnya sendiri melalui sebuah cerita bagaimana sebuah sistem dipakai. Urutan langkah- langkah yang menerangkan antara pengguna dengan sistem disebut *scenario*. Setiap *scenario* mendeskripsikan urutan kejadian. Setiap urutan diinisialisasi oleh orang, sistem yang lain, perangkat keras atau urutan waktu. Dengan demikian secara singkat, bisa dikatakan *use case* adalah serangkaian *scenario* yang digabungkan bersama- sama oleh tujuan umum pengguna.

Setelah membuat *use case* diagram, selanjutnya adalah membuat *class* diagram. *Class* diagram mendeskripsikan jenis-jenis objek dalam sistem dan berbagai macam hubungan statis yang terdapat diantara *class*- *class* tersebut. *Class* diagram juga menunjukkan *property* dan operasi sebuah *class* dan batasan-batasan yang terdapat dalam hubungan objek tersebut, namun sebelum membuat *class* diagram harus ditentukan terlebih dahulu *entity class* yang ada pada sistem.

*Activity* diagram memungkinkan siapapun yang melakukan proses untuk memilih urutan dalam melakukan. Dengan kata lain, diagram hanya menyebutkan aturan- aturan rangkaian dasar yang harus kita ikuti. Hal ini penting untuk pemodelan bisnis karena proses- proses sering muncul secara paralel. Ini juga berguna pada algoritma yang bersamaan, dimana urutan- urutan independent dapat melakukan hal- hal secara paralel.

Activity diagram menggambarkan berbagai alur aktivitas dalam sistem yang sedang dirancang, bagaimana masing-masing alur berawal, decision yang mungkin terjadi, dan bagaimana mereka berakhir. Activity diagram juga dapat menggambarkan proses paralel yang mungkin terjadi pada beberapa eksekusi. Activity diagram merupakan state diagram khusus, di mana sebagian besar state adalah action dan sebagian besar transisi di-trigger oleh selesainya state sebelumnya (internal processing). Oleh karena itu activity diagram tidak menggambarkan behaviour internal sebuah sistem (dan interaksi antar subsistem) secara eksak, tetapi lebih menggambarkan proses-proses dan jalur-jalur aktivitas dari level atas secara umum. Menggambarkan proses bisnis dan urutan aktivitas dalam sebuah proses. Dipakai pada business modeling untuk memperlihatkan

urutan aktifitas proses bisnis. Struktur diagram ini mirip flowchart atau Data Flow Diagram pada perancangan terstruktur. Sangat bermanfaat apabila kita membuat diagram ini terlebih dahulu dalam memodelkan sebuah proses untuk membantu memahami proses secara keseluruhan. Activity diagram dibuat berdasarkan sebuah atau beberapa use case pada use case diagram.

Sequence diagram (diagram urutan) adalah suatu diagram yang memperlihatkan atau menampilkan interaksi-interaksi antar objek di dalam sistem yang disusun pada sebuah urutan atau rangkaian waktu. Interaksi antar objek tersebut termasuk pengguna, display, dan sebagainya berupa pesan/message. Sequence Diagram digunakan untuk menggambarkan skenario atau rangkaian langkah-langkah yang dilakukan sebagai sebuah respon dari suatu kejadian/event untuk menghasilkan output tertentu. Sequence Diagram diawali dari apa yang me-trigger aktivitas tersebut, proses dan perubahan apa saja yang terjadi secara internal dan output apa yang dihasilkan.

Diagram ini secara khusus berasosiasi dengan use case diagram. Sequence diagram juga memperlihatkan tahap demi tahap apa yang seharusnya terjadi untuk menghasilkan sesuatu didalam use case. Sequence diagram biasanya digunakan untuk tujuan analisa dan desain, memfokuskan pada identifikasi method didalam sebuah system. Sequence diagram biasanya dipakai untuk memodelkan :

- Deskripsi tentang system yang ada pada sebuah / beberapa use case pada use case diagram, yang menggambarkan hubungan antara actor dan use case diagram.
- Logika dari method (operation, function atau procedure).
- Logika dari service (high level method).

*Sequence* diagram menggambarkan interaksi antar objek di dalam dan di sekitar sistem (termasuk pengguna, *display*, dan sebagainya) berupa *message* yang digambarkan terhadap waktu. *Sequence* diagram biasa digunakan untuk menggambarkan skenario atau rangkaian langkah-langkah yang dilakukan sebagai *respons* dari sebuah event untuk menghasilkan *output* tertentu.

### 3.2.4 Pengujian Software

Menurut Roger S. Pressman (2002 : 525) dalam bukunya *Software Engineering : A Practioners's Approach* mendefinisikan pengujian *software* (perangkat lunak) adalah elemen kritis dari jaminan kualitas perangkat lunak dan merepresentasikan kajian pokok dari spesifikasi, desain, dan pengkodean.

Dalam buku klasiknya mengenai pengujian perangkat lunak, Glen Myers menyatakan sejumlah aturan yang berfungsi sebagai sasaran pengujian pada perangkat lunak adalah:

1. Pengujian adalah proses eksekusi suatu program dengan maksud menemukan kesalahan
2. *Testcase* yang baik adalah *testcase* yang memiliki probabilitas tinggi untuk menemukan kesalahan yang belum pernah ditemukan sebelumnya

3. Pengujian yang sukses adalah pengujian yang mengungkap semua kesalahan yang belum pernah ditemukan sebelumnya

Sasaran kita adalah mendesain pengujian yang secara sistematis mengungkap kelas kesalahan yang berbeda dan melakukan dengan jumlah waktu dan usaha minimum. Terdapat dua teknik pendekatan dalam melakukan pengujian *software*, yaitu :

1) **Pengujian *White-Box***

Pengujian ini berfokus pada struktur kontrol program. *Testcase* dilakukan untuk memastikan bahwa semua *statemen* pada program telah dieksekusi paling tidak satu kali selama pengujian dan bahwa semua kondisi logis telah diuji.

2) **Pengujian *Black-Box***

Pengujian ini 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, metode ini melakukan pengujian terhadap fungsi operasional *software.*

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 *database* eksternal,
4. Kesalahan kinerja,

5. Inisialisasi dan kesalahan terminasi.

Teknik atau metode pengujian *software* yang penulis akan gunakan adalah pengujian *black-box*. Pengujian *black-box* menyinggung ujicoba yang dilakukan pada *interface software* (GUI). Apakah *input* diterima dengan benar, dan *output* yang dihasilkan benar, apakah integritas informasi eksternal terpelihara. Ujicoba *black-box* memeriksa beberapa aspek sistem, tetapi memeriksa sedikit mengenai struktur *logical internal software.*