BAB III

OBJEK DAN METODE PENELITIAN

3.1. Objek Penelitian

Untuk penyusunan skripsi ini, penulis melakukan penelitian mengenai Sistem Informasi pembelian dan Penjualan di CV.Kurnia Jaya.

3.1.1. Sejarah Singkat CV.Kurnia Jaya

CV.Kurnia Jaya Putra didirikan sejak tahun 1972. Pada awal berdiri bernama CV. Kurnia Jaya namun di tahun 2008 berubah nama menjadi CV.Kurnia Jaya Putra yang juga beralihnya kepemipinan dan sebagian saham yang pada awal pendirianya sepenuhnya berasal dari modal pribadi pemilik dari CV.Kurnia Jaya. Perusahaan yang bergerak dibidang usaha jual beli gula merah merupakam perusahaan yang sedang berkembang dengan jumlah supplier yang terus bertambah dari tahun ke tahun.

3.1.2. Visi dan Misi CV. Kurnia Jaya

Visi

Melakukan kegiatan usaha dengan dasar kejujuran dan prinsip adil serta meningkatakan omzet perusahaan dengan cara-cara yang halal dan tidak menimbulkan benturan dengan pihak manapun

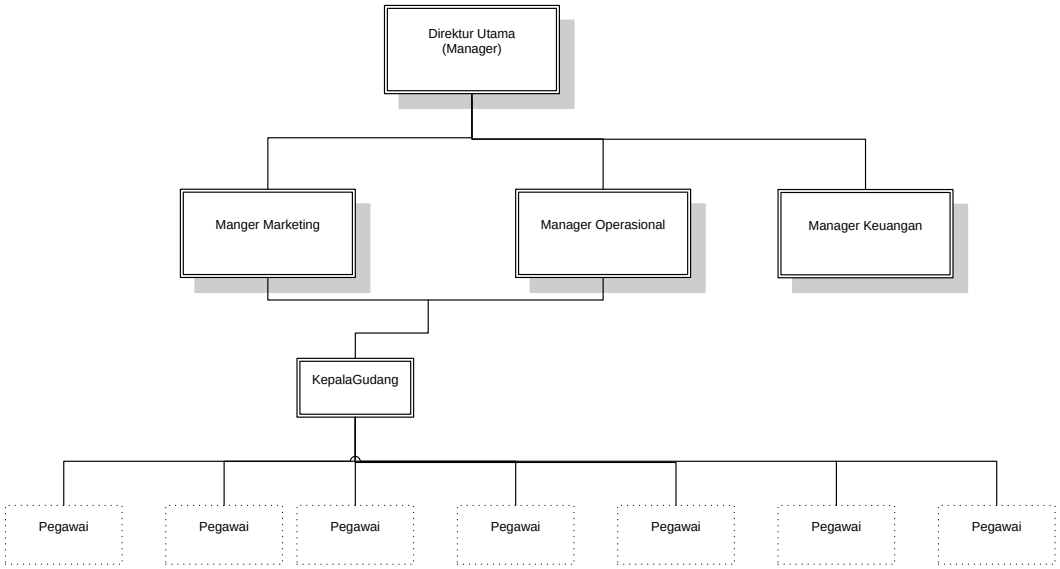
Misi

Misi dari CV. Kurnia Jaya adalah:

a. Melakukan pengembangan usaha penjualan dan pembelian di bidang gula merah.
b. Menjadi mitra terpercaya baik bagi supplier maupun buyer
c. Meningkatkan kesejahteraan semua pihak yang terkait dengan perusahaan baik untuk pemilik, pekerja dan juga para supplier.

3.1.3. Struktur Organisasi CV. Kurnia Jaya

Didalam CV.Kurnia Jaya terdapat sebuah struktur organisasi sebagai berikut



Gambar 3.1 Struktur Organisasi Perusahaan

(sumber CV.Kurnia Jaya)

1. Direktur Utama (Manager Utama)

   Seseorang yang ditunjuk untuk memimpin perusahaan. Direktur dapat seseorang yang memiliki perusahaan tersebut atau orang

professional yang ditunjuk oleh pemilik usaha untuk menjalankan dan meminpin perusahaan. Penyebutan direktur dapat bermacam-macam yaitu dewan manager, dewan gubernur, atau dewan eksekutif.

2. Manajer Marketing

   Seseorang yang meiliki keahlian atau pengetahuan tentang pemasaran, untuk mempertahankan kelangsungan perusahaanya, untuk berkembang dan untuk mendapatkan laba. Kegiatan manager pemsaran harus memberikan kepuasan yang baik kepada konsumen jika ingin usahanya berjalan terus.

3. Manajer Keuangan

   Seseorang didalam perusahaan yang harus mengetahui bagaimana cara mengelola segala unsur dari segi keuangan, hal ini wajib dilakukan karena keuangan merupakanslah satu fungsi penting dalam mencapai tujuan perusahaan. Seorang manager keuangan harus mampu mengetahui segala aktivitas manajemenkeuangan, khuussnya penganalisisan sumber dana dan penggunaanya untuk merealisasikan keuntungan maksimum bagi perusahaan tersebut. Seorang manager keuangan harus memahami arus peredaran uang baik eksternal maupun internal.

4. Manager Operasional

   Manager Operasional bertanggung jawab kepada Direktur Utama.

3.1.4. Deskripsi Tugas

a) Tugas Direktur Utama

1. Meminpin perusahaan dengan menerbitkan kebijakan-kebijakan perusahaan.
2. Memilih, menetapkan, mengawasi tugas dari karyawan dan kepala bagian (*manager*).
3. Menyetujui anggran tahunan perusahaan

b) Manajer Marketing

Tugasnya antara lain menanalisa, merencanakan, melaksanakan, mengawasi program-program yang bertujuan untuk menimbulkan pertukaran dengan pasar yang dituju dengan maksud untuk mencapai tujuan perusahaan (Kotler,1980)

c) Tugas Manajer Keuangan

1. Melakukan pengawasan biaya
2. Melakukan kebijakan harga
3. Meramalkan laba yang akan datang
4. Mengukur biaya modal kerja.

d) Tugas Direktur Operasional

1. Menjaga kontinuitas produk
2. Mengikuti perkembangan produksi agar dapat diadakan peningkatan produksi yang lebih baik dalam penyempurnaan produksi.

3.2. Metode Penelitian

3.2.1.Desain Penelitian

Agar penelitian berjalan sesuai dengan yang diharapkan maka perlu ada perencanaan yang matang. Penulis melakukan design terlebih dahulu tentang metode penelitian yang meliputi:

Alat pengukuran data, waktu dan tempat pengambilan data dan metode analisa data.

1. Tempat pengambilan data

   Penelitian yang dilakukan membutuhkan tempat *field* (lingkungan terbuka) atau *laboratory* (lingkungan terkontrol)

2. Waktu pengumpulan data

   Apakah pengambilan data membutuhkan waktu temporer atau terus menerus.

3.2.2. Jenis dan Metode Pengumpulan Data

Untuk memperoleh hasil pembahasan dari hasil masalah dan kesimpulan, diperlukan data dan informasi yang objectif. Adapun meode penelitian, mengumpulkan data dari informasi yang diperleh dalam menyusun proposal ini.

3.2.2.1.Sumber Data Primer

a. Wawancara

   Dengan cara melakukan tanya jawab secara langsung dengan para staf yang menangani masalah yang dibahas. Dan melakukan pengamatan langsung terhadap kegiatan

yang sedang berjalan sehingga penulis dapat melihat secara langsung cara kerja sistem di CV.Kurnia Jaya Putra

b. Observasi

Mengidentifikasi masalah yang ada di CV.Kurnia Jaya Putra dengan cara menentukan topik yang akan dibahas dalam proposal ini.

3.2.2.1. Sumber Data Sekunder

Di dalam pembuatan sistem informasi pembelian dan penjualan ini penulis juga menggunakan data sekunder sebagai acuan yaitu data yang diperoleh dari studi kepustakaan dengan cara mengumpulkan dan mempelajari data/informasi serta pengetahuan yang berasal dari buku-buku, literature, dokumen yang ada hubungannya dengan objek yang diteliti selain itu pula penulis melakukan penarian data / informasi melalui internet.

3.2.3. Metode Pendekatan dan Pengembangan Sistem

3.2.3.1. Metode Pendekatan Sistem

Metode pendekatan yang digunakan penulis yaitu metode pendekatan perancangan terstruktur sedangkan metode pengembangan sistem menggunakan metode

pengembangan sistem prototype. Untuk lebih jelasnya akan dijelaskan dibaawah ini.

Dalam pendekatan sistem penulis melakukan perancangan terhadap web bisnis yang penulis buat. Pendekatan sistem ini bertujuan untuk menghasilkan perancangan sistem informasi pemesanan, login, pendaftaran dan yang lainnya secara gambaran umum.

Metode pendekatan sistem ini berorientasi data (data oriented) dimana penulis mengumpulkan data-data yang diperlukan dalam perancangan sistem informasi yang ada di web bisnis, selain itu penulis menggunakan metode analisa dan perancangan terstruktur untuk sistem yang dibuat.

Perancangan sistem ini meliputi digram alir dokumen, diagram kontek dan diagram alir data. Dalam tahap perancangan ini penulis menggambarkan mengenai aliran informasi dan proses yang sedang berjalan dalam sistem.

3.2.3.2. **Metode Pengembangan Sistem**

Model *Prototyping*

Metode ini sering digunakan pada dunia riil. Karena metode ini secara keseluruhan akan mengacu kepada kepuasan

user. Bisa dikatakan bahwa metode ini merupakan metode waterfall yang dilakukan secara berulang-ulang.

Tahapan Metode *Prototyping*

1. Survey.
2. Pengumpulan data
3. Analisa data dan sistem .
4. Perancangan sistem
5. Implementasi Sistem
6. Evaluasi dan laporan

Jenis-jenis *Prototyping*

1) *Feasibility prototyping.* Digunakan untuk menguji kelayakan dari teknologi yang akan digunakan untuk sistem informasi yang akan disusun.
2) *Requirement prototyping*. Digunakan untuk mengetahui kebutuhan aktivitas bisnis user. Misalnya dalam sebuah perusahaan terdapat user direktur, manajer, dan karyawan. Maka penggunaan sistem dapat dibedakan berdasarkan user tersebut sesuai dengan kebutuhannya.
3) *Design Prototyping*. Digunakan untuk mendorong perancangan sistem informasi yang akan digunakan.

4) *Implementation prototyping*. Merupakan lanjutan dari rancangan protipe, prototype ini langsung disusun sebagai suatu sistem informasi yang akan digunakan. (www.informaticse.com)

3.2.3.3. **Alat Bantu Analisis dan Perancangan**

1) Flow Map

Flow map adalah bagan alir sistem yang menunjukan arus pekerjaan secara keseluruhan dari simbol-simbol untuk menggambarkan secara urut dari arus data dan dokumen baik yang diperlukan maupun yang dihasilkan.

2) Diagram Kontek

Diagram konteks merupakan alat pemodelan atau suatu diagram yang menggambarkan sistem berbasis komputer yang dirancang secara global dan merupakan suatu diagram alir data tingkat atas, dimana didalam diagram konteks ini menggambarkan seluruh jaringan, baik masukan maupun sebuah keluaran sebuah sistem.

Diagram konteks terdiri dari sebuah simbol proses tunggal yang menggambarkan sebuah sistem dan menunjukan data aliran utama untuk dan dari terminator. Diagram ini merupakan dasar yang digunakan untuk menentukan aliran data yang mengalir menuju sistem (*input system*) dan keluar dari sistem (*output system*), yang meliputi objek berupa kesatuan luar (*ekseternal entity*). Diagram konteks dapat mendefinisikan jangkauan proses

penurunan sistem informasi yaitu menetukan apa yang menjadi bagian dari sistem informasi dan apa yang tidak menjadi bagian sistem informasi.

3) *Data Flow Diagram*

Diagram aliran data atau DFD merupakan suatu bagan alir data yang digunakan untuk menjelaskan data yang ditransformasikan oleh suatu proses pada suatu sistem dengan menekankan pada fungsi – fungsi yang ada dalam sistem, cara menggunakan informasi yang tersimpan dan pemindahan informasi antar fungsi di dalam sistem.

Beberapa simbol yang digunakan dalam *Data Flow Diagram* (DFD) antara lain:

i. Proses (*Process*). Proses adalah simbol pertama data flow diagram. Proses dilambangkan dengan lingkaran, dimana proses ini menunjukan bagian dari sistem yang mengubah satu atau lebih input dan output. Nama proses dituliskan dengan satu kata, singkatan atau kalimat sederhana.

ii. Aliran Data (*Flow*)**.** Aliran Data digambarkan dengan tanda panah dari proses. Aliran data juga digunakan untuk menunjukan bagian – bagian informasi dari satu bagian ke bagian lain. Pembagian nama untuk aliran ini menunjukan sebuah arti untuk sebuah aliran. Untuk kebanyakan sistem

yang dibuat, aliran data sebenarnya mengambarkan data yakni angka, huruf, pesan, dan macam - macam informasi lainnya.

iii. Simpanan Data (*Storage*)**.** Simpanan data digunakan sebagai penyimpanan bagi paket – paket data. Notasi penyimpanan data digambarkan dengan garis horizontal yang pararel. Simpanan data merupakan simpanan data dari data yang berupa suatu file atau database di sistem komputer ataupun berupa arsip atau catatan manual. Nama dari simpanan data menunjukan nama filenya.

iv. Kesatuan Luar (*External Entity*)**.** Setiap sistem pasti mempunyai batas sistem (*boundary*) yang memisahkan suatu sistem dengan lingkungan luarnya. Kesatuan luar merupakan kesatuan (*entity*) dilingkungan luar sistem yang dapat berupa orang, Organisasi atau sistem lainnya yang berada di lingkungan luarnya yang akan memberikan input atau output dari sistem.

4) Kamus Data

Kamus data adalah *catalog* fakta tentang data dan kebutuhan-kebuthan informasi dari suatu system. Atau kumpulan daftar elemen data yang mengalir pada system perangkat lunak sehingga masukan (*input*) dan keluaran (*output*) dapat dipahami secara umum.

5) Perancangan Basis data

a. Normalisasi

Normalisasi suatu proses untuk mengubah suatu tabel yang memiliki masalah tertentu ke dalam dua buah tabel atau lebih, yang tidak lagi memiliki masalah tersebut (Abdul Kadir, 2002: 52).

b. Tabel Relasi

Relasi Antar Tabel adalah bagan yang menggambarkan hubungan antar tabel satu dengan tabel yang lain sehingga membentuk Basis Data.

Dalam tabel relasi terdapat hubungan antar tabel yang biasa disebut kardinalitas / derajat relasi. Kardinalitas terdiri dari 3 jenis diantaranya :

1. *One to one*

   One to one mempunyai pengertian setiap entitas A berhubungan dengan paling banyak dengan satu entitas pada himpunan entitas B, begitu pula sebaliknya setiap entitas pada hubungan entias B berhubungan dengan paling banyak dengan satu entitas pada himpunan entitas A.

2. *One to many*

   One to many mempunyai pengertian bahwa setiap entitas A dapat berhubungan dengan banyak entitas pada

himpunan entitas B, tetapi tidak sebaliknya, dimana setiap entitas pada hubungan entitas B berhubungan dengan paling banyak satu entitas pada himpunan entitas A.

3. *Many to one*

   Many to one mempunyai pengertian bahwa setiap entitas A berhubungan dengan paling banyak entitas B, tetapi tidak sebaliknya, dimana setiap entitas pada himpunanentitas A berhubungan dengan paling banyak entitas pada himpunan B.

4. *Many to mamy*

   Many to many mempunyai pengertian bahwa setiap entitas pada himpunan entitas A dapat berhubungan dengan banyak entitas pada himpunan entitas B, dan demikian juga sebaliknya, dimana setiap entitas pada himpunan entitas B dapat berhubungan dengan banyak entitas pada himpunan entitas A. (Fathansyah.2004)

3.2.4.Pengujian Software

Adalah elemen kritis dari jaminan kualitas perangkat lunak dan merepresentasikan kajian pokok dari spesifikasi, desain, dan pengkodean.

Pentingnya pengujian perangkat lunak dan implikasinya yang mengacu pada kualitas perangkat lunak

tidak dapat terlalu ditekan karena melibatkan sederetan aktivitas produksi di mana peluang terjadinya kesalahan manusia sangat besar dan arena ketidakmampuan manusia untuk melakukan dan berkomunikasi dengan sempurna maka pengembangan perangkat lunak diiringi dengan aktivitas jaminan kualitas.

Meningkatnya *visibilitas* (kemampuan) perangkat lunak sebagai suatu elemen sistem dan "biaya" yang muncul akibat kegagalan perangkat lunak, memotivasi dilakukannya perencanaan yang baik melalui pengujian yang teliti. Pada dasarnya, pengujian merupakan satu langkah dalam proses rekayasa perangkat lunak yang dapat dianggap sebagai hal yang merusak daripada membangun.

Sejumlah aturan yang berfungsi sebagai sasaran pengujian pada perangkat lunak adalah:

1. Pengujian adalah proses eksekusi suatu program dengan maksud menemukan kesalahan
2. Test case yang baik adalah test case 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 itu berlawanan dengan pandangan yang biasanya dipegang yang menyatakan bahwa pengujian yang berhasil adalah pengujian yang tidak ada kesalahan yang ditemukan. Data yang dikumpulkan pada saat pengujian dilakukan memberikan indikasi yang baik mengenai reliabilitas perangkat lunak dan beberapa menunjukkan kualitas perangkat lunak secara keseluruhan, tetapi ada satu hal yang tidak dapat dilakukan oleh pengujian, yaitu pengujian tidak dapat memperlihatkan tidak adanya cacat, pengujian hanya dapat memperlihatkan bahwa ada kesalahan perangkat lunak.

Sebelum mengaplikasikan metode untuk mendesain *test case* yang efektif, perekayasa perangkat lunak harus memahami prinsip dasar yang menuntun pengujian perangkat lunak, yaitu:

1. Semua pengujian harus dapat ditelusuri sampai ke persyaratan pelanggan, maksudnya mengungkap kesalahan dari cacat yang menyebabkan program gagal.
2. Pengujian harus direncanakan lama sebelum pengujian itu mulai, maksudnya semua pengujian dapat direncanakan dan dirancang sebelum semua kode dijalankan.
3. Prinsip Pareto berlaku untuk pengujian perangkat lunak, maksudnya dari 80% kesalahan yang ditemukan selama

pengujian dapat ditelusuri sampai 20% dari semua modul program.

4. Pengujian harus mulai "dari yang kecil" dan berkembang ke pengujian "yang besar", Selagi pengujian berlangsung maju, pengujian mengubah *focus* dalam usaha menemukan kesalahan pada cluster modul yang terintegrasi dan akhirnya pada sistem.
5. Pengujian yang mendalam tidak mungkin karena tidak mungkin mengeksekusi setiap kombinasi jalur skema pengujian dikarenakan jumlah jalur permutasi untuk program menengah pun sangat besar.
6. Untuk menjadi paling efektif, pengujian harus dilakukan oleh pihak ketiga yang independent

Dalam lingkungan yang ideal, perekayasa perangkat lunak mendesain suatu program komputer, sebuah sistem atau produk dengan testabilitas dalam pikirannya. Hal ini memungkinkan individu yang berurusan dengan pengujian mendesain test case yang efektif secara lebih mudah. Testabilitas adalah seberapa mudah sebuah program komputer dapat diuji. Karena sangat sulit, perlu diketahui apa yang dapat dilakukan untuk membuatnya menjadi lebih mudah. Procedural dan menggunakannya sebagai pedoman untuk menetapkan basis set dari jalur eksekusi.

Sasaran utama desain *test case* adalah untuk mendapatkan serangkaian pengujian yang memiliki kemungkinan tertinggi di dalam pengungkapan kesalahan pada perangkat lunak. Untuk mencapai sasaran tersebut, digunakan 4 kategori yang berbeda dari tehnik desain test case: Pengujian *white-box*, pengujian *black-box*, *Integrasi Bottom-Up* dan *Integrasi Top-Down.*

Pengujian *white-box* berfokus pada struktur control program. *Test case* dilakukan untuk memastikan bahwa semua statemen pada program telah dieksekusi paling tidak satu kali selama pengujian dan bahwa semua kondisi logis telah diuji. Pengujian basic path, tehnik pengujian *white-box*, menggunakan grafik (matriks grafiks) untuk melakukan serangkaian pengujian yang independent secara linear yang akan memastikan cakupan.

Pengujian aliran data dan kondisi lebih lanjut menggunakan logika program dan pengujian loop menyempurnakan tehnik *white-box* yang lain dengan memberikan sebuah prosedur untuk menguji loop dari tingkat kompleksitas yang bervariasi. Pengujian *black-box* didesain untuk mengungkap kesalahan pada persyaratan fungsional tanpa mengabaikan kerja internal dari suatu program.

Teknik pengujian *black-box* berfokus pada domain informasi dari perangkat lunak, dengan melakukan *test case* dengan

menpartisi domain input dari suatu program dengan cara yang memberikan cakupan pengujian yang mendalam.

Metode pengujian *graph-based* mengeksplorasi hubungan antara dan tingkah laku objek-objek program. Partisi ekivalensi membagi domain input ke dalam kelas data yang mungkin untuk melakukan fungsi perangkat lunak tertentu. Analisis nilai batas memeriksaa kemampuan program untuk menangani data pada batas yang dapat diterima.

Metode pengujian yang terspesialisasi meliputi sejumlah luas kemampuan perangkat lunak dan area aplikasi. GUI, arsitektur *client/ server*, dokumentasi dan fasilitas *help* dan sistem *real time* masing-masing membutuhkan pedoman dan tehnik khusus untuk pengujian perangkat lunak.

Integrasi Top-Down adalah pendekatan incremental dengan menggerakkan ke bawah melalui hirarki control, dimulai dengan control utama. Strategi intergrasi *top-down* memeriksa control mayor atau keputusan pada saat awal di dalam proses pengujian. Pada struktur program yang difaktorkan dengan baik, penarikan keputusan terjadi pada tingkat hirarki yang lebih tinggi sehingga terjadi lebih dulu.

Strategi *top-down* kelihatannya tidak sangat rumit, tetapi di dalam praktenya banyak menimbulkan masalah logistic. Biasanya

masalah ini terjadi jika dibutuhkan pemrosesan di dalam hirarki pada tingkat rendah untuk menguji secara memadai tingkat yang lebih tinggi.

Pengujian Integrasi *Bottom-up* memulai konstruksi dan pengujian dengan modul atomic (modul pada tingkat paling rendah pada struktur program). Karena modul diintegrasikan dari bawah ke atas, maka pemrosesan yang diperlukan untuk modul subordinate ke suatu tuingkat yang diberikan akan selalu tersedia dan kebutuhan akan stub dapat dieliminasi. Strategi integrasi bottom-up dapat diimplementasi dengan langkah-langkah:

1. Modul tingkat rendah digabung ke dalam cluster (build) yang melakukan subfungsi perangkat lunak spesifik.
2. Driver (program control untuk pengujian) ditulis untuk mengkoordinasi input dan output test case
3. *Cluster* diuji
4. *Driver* diganti dan *cluster* digabungkan dengan menggerakkannya ke atas di dalam struktur program.