BAB III

OBJEK DAN METODE PENELITIAN

3. 1 Objek Penelitian

Dalam penelitian ini, yang menjadi objek penelitian adalah CV. Panca Mandiri, Jl. Cibadak Gg. Citepus No.4 kelurahan cibadak, kecamatan astanaanyar, Rt02 Rw03 Bandung.

3.1.1. Sejarah Singkat Perusahaan

CV. PANCA MANDIRI adalah sebuah home industri yang terdiri dari lima jenis home industri.Pada tahun 1993 home industri ini berdiri, begerak dari home industri kecil hingga pada tahun 1998 berkembang menjadi CV.

Pabrik roti ini saat ini memiliki karyawan yang berjumlah 12 orang. Jam operasional pabrik ini dimulai dari pukul 9.00 wib sampai pukul 17.00 wib. Produk utama dari pabrik ini adalah roti basah, roti kering, roti tawar dan roti isi.

Offset dan printing, pembuatan dan penjualan kecap, pembuatan dan penjualan Roti, steel, furniture. Dari ke lima jenis perusahaan di atas merupakan satu kesatuan dari CV. PANCA MANDIRI.

3.1.2. Visi dan Misi Perusahaan

a. Visi

Visi dari pada CV. PANCA MANDIRI :

“Visi dari perusahaan ini adalah meraih keberhasilan untuk mencapai kepuasan, memberikan kepuasan kepada konsumen dengan cara meningkatkan kualitas

produk yang kami pasarkan kepada konsumen. Kepuasan konsumen berarti kepuasan perusahaan. ”.

b. Misi

Adapun yang menjadi misi CV. PANCA MANDIRI diantaranya :

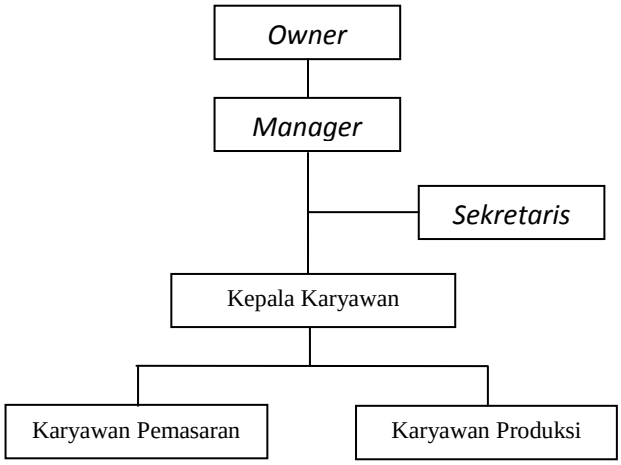
1. Meningkatkan kualitas produk kami dengan tujuan yang lebih untuk jangka panjang.
2. Menjadikan produk sebagai konsumsi masyarakat yang mudah dijangkau dan didapatkan.
3. Memproduksi produk – produk yang berkualitas baik dan mendapatkan area pemasaran yang terus meluas.
4. Mengembangkan inovasi terhadap promosi produk – produk yang perusahaan buat seperti pengadaan Sistem Informasi penjualan roti berbasis web.

3.1.3. Struktur Organisasi Perusahaan

Struktur organisasi pada perusahaan sangatlah menentukan kelancaran jalannya perusahaan, dimana disini hubungan antara pimpinan dengan bawahan diatur dengan sebaik-baiknya dan dengan struktur organisasi perusahaan dapat mengatur atau menjelaskan wewenang pekerjaan masing-masing fungsi atau seksi agar tidak terjadi kesalahpahaman pekerjaan. Jadi jelas bahwa struktur organisasi perusahaan merupakan hal yang sangat penting untuk mencapai sistem kerja yang baik.

Untuk lebih jelasnya struktur organisasi CV. Panca Mandiri dapat dilihat pada gambar 3.1. Tugas dari masing-masing bagian sesuai dengan kedudukannya di dalam perusahaan tersebut adalah sebagai berikut:

STRUKTUR ORGANISASI CV. PANCA MANDIRI



Gambar 3.1 Struktur Organisasi CV. Panca Mandiri

3.1.4. Deskripsi Tugas

Adapun deskripsi tugas dari masing-masing jabatan tersebut adalah :

1. *Owner*

   Adalah pemegang saham secara keseluruhan dan mempunyai kewenangan penuh atas segala bentuk kebijakan perusahaan.

2. *Manager*

   Manager sebagai atasan yang berada di bawah kekuasaan / kebijakan direktur perusahaan, mempunyai tugas sebagai berikut:

   a. Mengatur kegiatan – kegiatan apa yang harus dilakukan.
   b. Mengatur karyawan – karyawan yang menjalankan kegiatan – kegiatan perusahaan.
   c. Memantau secara langsung pengerjaan kegiatan atau pelaksanaan rencana kerja, apakah sesuai dengan aturan – aturan umum dalam perusahaan atau tidak.

3. *Sekretaris*

   Membantu *manager* menyusun program kerja, serta mencakup dalam kontrol keuangan serta menerima segala bentuk pembayaran pada tugas kasir. Mencakup juga sebagai admin.

4. Kepala Karyawan

   Berfungsi sebagai mengontrol pekerjaan setiap karyawanpada tugasnya masing-masing.

5. Karyawan Pemasaran

   Karyawan ini lebih identik dengan sebutan kurir karena memiliki tugas memasarkan atau mengirimkan produk perusahaan langsung kepada pihak distributor.

6. Karyawan Produksi

   Memproduksi hasil barang yang di buat sesuai target yang di tetapkan oleh *manager*.

3.2. Metode Penelitian

Penelitian adalah kegiatan untuk memperoleh fakta-fakta atau prinsip-prinsip (baik kegiatan untuk penemuan, pengujian atau pengembangan) dari suatu pengetahuan dengan cara mengumpulkan, mencatat dan menganalisa data yang dikerjakan secara sistematis berdasarkan ilmu pengetahuan (metode ilmiah).

Metode yang digunakan dalam melaksanakan penelitian merupakan dasar penyusunan rancangan penelitian dan merupakan penjabaran dari metode ilmiah secara umum.

3.2.1. Desain Penelitian

Desain penelitian ini termasuk kedalam penelitian yang bersifat deskriptif. Penelitian deskriptif adalah penelitian yang bertujuan untuk memperoleh ciri-ciri variabel, dimana dalam penelitian ini adalah untuk memperoleh gambaran tentang kinerja program dirancang dan diimplementasikan kepada pengguna (user) dalam perusahaan.

3.2.2. Jenis dan Metode Pengumpulan Data

3.2.2.1. Sumber Data Primer

Sumber data/informasi penelitian ini berdasarkan kepada jenis data yang diperlukan. Data primer yaitu data yang diperoleh dari responden secara langsung yang dikumpulkan melalui *survey* lapangan dengan menggunakan teknik pengumpulan data tertentu yang dibuat khusus untuk itu. Teknik pengumpulan data dalam rangka pengumpulan informasi mengenai objek penelitian ini, yaitu :

1. Metode Observasi

   Dengan mengadakan pengamatan langsung kelapangan yang disertai dengan studi literatur yang dapat memperkuat tujuan yang hendak dicapai.

2. Riset Lapangan (*Field Research)*

   Data-data yang diambil secara langsung pada objek permasalahan yang berkaitan dengan penelitian ini adalah melalui :

   a. Wawancara (interview) merupakan suatu teknik pengumpulan data dengan cara bertatap muka antara pengumpul data dan nara sumber yang dimaksud.

Wawancara dilakukan secara langsung dengan memberikan beberapa daftar pertanyaan yang telah disiapkan.

b. Studi Pustaka untuk mengumpulkan berbagai referensi dari buku-buku menunjang dalam pengolahan data yang didapatkan dari perpustakaan universitas maupun perusahaan yang terkait, dan membaca pada *website* di internet yang membahas secara umum mengenai hal-hal yang berhubungan dengan judul yang diangkat oleh penulis.

3.2.2.2. Sumber Data Sekunder

Dilakukan dengan menelaah data sekunder yang digunakan untuk menunjang, melengkapi dan menyempurnakan data primer, diperoleh dari dokumentasi perusahaan, kepustakaan, internet, dan data-data dari gudang, serta hasil penelitian sebelumnya yang berhubungan dengan masalah dalam penelitian ini.

3.2.3. Metode Pendekatan dan Pengembangan Sistem

Dalam sub bab ini akan dijelaskan mengenai metode pendekatan, pengembangan sistem, dan alat bantu analisis perancangan. Berikut ini adalah uraian dari metode pendekatan dan pengembangan sistem :

3.2.3.1. Metode Pendekatan Sistem

Metode pendekatan sistem yang digunakan penulis adalah dengan menggunakan pendekatan berorientasi objek. Pendekatan berorientasi objek adalah cara memandang persoalan menggunakan model-model yang diorganisasikan seputar konsep objek yang mengkombinasikan struktur data dan prilaku entitas, hal ini lebih efektif karena objek-objek dapat mempresentasikan

bagian-bagian dari dunia eksternal, mempersempit kesenjangan (*gap*) konseptual antara dunia eksternal dan komponen-komponen perangkat lunak. Alat bantu yang digunakan berupa UML (*Unified Modelling Language*) untuk melakukan OOA (*Object Oriented Analysis*) dan OOD (*Object Oriented Design*).

UML (*Unified Modelling Language*) merupakan bahasa standar untuk pengembangan sebuah *software* yang dapat menyampaikan bagaimana membuat dan membentuk model-model, tetapi tidak menyampaikan apa dan kapan model yang seharusnya dibuat yang merupakan salah satu proses implementasi pengembangan *software*.

3.2.3.2. Metode Pengembangan Sistem

Metode penelitian merupakan suatu teknik atau cara untuk memperoleh data yang dibutuhkan untuk menyusun karya ilmiah, kemudian faktor apa saja yang terkait dengan pokok permasalahan yang ada. Guna menyelesaikan permasalahan yang telah di bahas maka dalam pengembangan sistem ini penulis menggunakan Metode *prototype.* Prototype adalah suatu proses yang memungkinkan pengembangan aplikasi untuk menciptakan suatu model dari sistem informasi yang harus dikembangkan. Tujuan utama dari prototype adalah untuk mengurangi resiko dan ketidakpastian selama tahap-tahap awal dari *life cyle* pengembangan sistem informasi.

Menurut Soendoro dan Haryanto (2005:188), terdapat beberapa keuntungan dalam menggunakan prototype pada proses perangkat lunak antara lain :

1. Mengurangi kesalahpahaman antara pengembang dan pemakai sistem informasi.
2. Contoh sistem bisa diperlihatkan kepemakai dalam tempo singkat.

3. *Prototype* dipakai sebagai dasar spesifikasi dari sistem.

Adapun Tahapan-tahapan dalam *Prototyping* adalah sebagai berikut:

1. Mengumpulkan data atau informasi dari *customer* mengenai kebutuhan objek yang diperlukan untuk perangkat lunak.
2. Membangun atau memperbaiki *prototype* perangkat lunak, dimana didalamnya dilakukan*"quick design"* (perancangan cepat) dan membuat *prototype* sesuai perancangan tersebut.
3. Melakukan evaluasi *prototype* yaitu dengan menunjukan *prototype* tersebut kepada *customer* untuk disesuaikan dengan keinginan *customer*.

3.2.3.3. Alat Bantu Analisis dan Perancangan

Unified Modelling Language(UML) adalah sebuah "bahasa" yg telah menjadi standar dalamindustri untuk visualisasi, merancang dan mendokumentasikan sistem piranti lunak. UML menawarkan sebuah standar untuk merancang model sebuah sistem. Dengan menggunakan UML, kita dapat membuat model untuk semua jenis aplikasi perangkat lunak, yang mana aplikasi tersebut dapat berjalan pada perangkat keras, sistem operasi dan model jaringan apapun. Penggunaan UML dalam membangun model suatu sistem yang berbentuk diagram-diagram memudahkan baik pengguna maupun programmer dalam memahami urutan dalam sebuah sistem yang dirancang. Bentuk diagram yang digunakan untuk merepresentasikannya adalah sebagai berikut :

1. *Use-case Diagram*
2. *Class Diagram*
3. *State Diagram*
4. *Sequence diagram*

5. *Collaboration Diagram*
6. *Activity Diagram*
7. *Component Diagram*
8. *Deployment Diagram*

1. *Use-case diagram*

Use-case diagram merupakan suatu bentuk diagram yang menggambarkan fungsi-fungsi yang diharapkan dari sebuah sistem yang dirancang. Dalam *Use-case diagram* penekanannya adalah "*apa*" yag diperbuat oleh sistem, dan bukan "*bagaimana*". Sebuah *use-case* akan merepresentasikan sebuah interaksi antara pelaku atau *actor* dengan sistem. *Use-case diagram* yang digunakan dalam merancang suatu sistem dapat sangat membantu pada saat kita menyusun requirement sebuah sistem, mengomunikasikannya dengan klien, dan merancang pengujian untuk semua fitur yang terdapat dalam sistem.

2. Class Diagram

Class adalah sebuah spesifikasi yang jika diinstansiasi akan menghasilkan sebuah objek danmerupakan inti dari pengembangan dan desain berorientasi objek. *Class* menggambarkan keadaan(atribut/properti) suatu sistem, sekaligus menawarkan layanan untuk memanipulasi keadaan tersebut(metoda/fungsi). *Class diagram* menggambarkan struktur dan deskripsi *class, package* dan objek beserta hubungansatu sama lain seperti *containment*, pewarisan, asosiasi, dan lain-lain.

Class memiliki tiga area pokok :

1. Nama (dan stereotype)
2. Atribut
3. Metoda

Atribut dan metoda dapat memiliki salah satu sifat berikut :

a. *Private*, tidak dapat dipanggil dari luar *class* yang bersangkutan
b. *Protected*, hanya dapat dipanggil oleh *class* yang bersangkutan dan anak-anak yang mewarisinya
c. *Public*, dapat dipanggil oleh siapa saja

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.

3. Statechart Diagram

Statechart diagram menggambarkan transisi dan perubahan keadaan (dari satu *state* ke *state* lainnya) suatu objek pada sistem sebagai akibat dari *stimuli* yang diterima. Pada umumnya *statechart diagram* menggambarkan *class* tertentu (satu *class* dapat memiliki lebih dari satu *statechart diagram*). Dalam UML, *state* digambarkan berbentuk segiempat dengan sudut membulat dan memiliki nama sesuai kondisinya saat itu. Transisi antar *state* umumnya memiliki kondisi *guard* yang merupakan syarat terjadinya transisi yang bersangkutan, dituliskan dalam kurung siku. *Action* yang dilakukan sebagai akibat dari *event* tertentu dituliskan dengan diawali garis miring.

4. Sequence diagram

Sequence Diagram merupakan diagram yang mengambarkan kolaborasi yang dinamis antara obyek satu dengan yang lain. Kolaborasi ini ditunjukkan dengan adanya interaksi antar obyek di dalam dan di sekitar sistem yang berupa pesan atau instruksi yang berurutan. *Sequence diagram* umumnya digunakan

untuk menggambarkan suatu skenario atau urutan langkah-langkah yang dilakukan baik oleh *actor* maupun sistem yang merupakan respon dari sebuah kejadian untuk mendapatkan hasil atau output.

5. Collaboration Diagram

Collaboration diagram juga menggambarkan interaksi antar objek seperti *sequence diagram*, tetapi lebih menekankan pada peran masing-masing objek dan bukan pada waktu penyampaian *message*. Setiap *message* memiliki *sequence number*, di mana *message* dari level tertinggi memiliki nomor 1. Messages dari level yang sama memiliki prefiks yang sama.

6. Activity Diagram

Sebuah *Activity Diagram* menunjukkan suatu alur kegiatan secara berurutan. *Activity Diagram* digunakan untuk mendiskripsikan kegiatan-kegiatan dalam sebuah operasi meskipun juga dapat digunakan untuk mendeskripsikan alur kegiatan yang lainnya seperti *use case* atau suatu interaksi. Sama seperti state, standar UML menggunakan segiempat dengan sudut membulat untuk menggambarkan aktivitas. Decision digunakan untuk menggambarkan behaviour pada kondisi tertentu. Untuk mengilustrasikan proses-proses paralel (fork dan join) digunakan titik sinkronisasi yang dapat berupa titik, garis horizontal atau vertikal.

7. Component Diagram

Component diagram menggambarkan struktur dan hubungan antar komponen piranti lunak,termasuk ketergantungan (*dependency*) di antaranya. Komponen piranti lunak adalah modul berisi *code*, baik berisi *source code* maupun *binary code*, baik *library* maupun *executable*, baik yang muncul pada

compile time, link time, maupun *run time*. Umumnya komponen terbentuk dari beberapa *class* dan/atau *package*, tapi dapat juga dari komponen-komponen yang lebih kecil. Komponen dapat juga berupa *interface*, yaitu kumpulan layanan yang disediakan sebuah komponen untuk komponen lain.

8. Deployment Diagram

Deployment/physical diagram menggambarkan detail bagaimana komponen di-*deploy* dalam infrastruktur sistem, di mana komponen akan terletak (pada mesin, server atau piranti keras apa), bagaimana kemampuan jaringan pada lokasi tersebut, spesifikasi server, dan hal-hal lain yang bersifat fisikal. Sebuah *node* adalah server, *workstation*, atau piranti keras lain yang digunakan untuk men-*deploy* komponen dalam lingkungan sebenarnya. Hubungan antar *node* (misalnya TCP/IP) dan *requirement* dapat juga didefinisikan dalam diagram ini.

3.2.4. Pengujian Software

Pengujian *Software* adalah cara atau teknik untuk menguji perangkat lunak, mempunyai mekanisme untuk menentukan data uji yang dapat menguji perangkat lunak secara lengkap dan mempunyai kemungkinan tinggi untuk menemukan kesalahan. Pada pengujian perangkat lunak penulis memilih pengujian black-box dimana 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 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 :

a. Fungsi-fungsi yang tidak benar atau hilang.
b. Kesalahan interface.
c. Kesalahan dalam struktur data atau akses database eksternal.
d. Kesalahan kinerja.
e. Inisialisasi dan kesalahan terminasi.

Tidak seperti pengujian white-box, yang dilakukan pada awal proses pengujian, pengujian black-box cenderung diaplikasikan selama tahap akhir pengujian. Karena pengujian black-box memperhatikan struktur kontrol, maka perhatian berfokus pada domain informasi. Pengujian didesain untuk menjawab pertanyaan-pertanyaan berikut:

a. Bagaimana validitas fungsional diuji?
b. Kelas input apa yang akan membuat test-case menjadi baik?
c. Apakah sistem sangat sensitif terhadap harga input tertentu?
d. Bagaimana batasan dari suatu data disolasi?
e. Kecepatan data apa dan volume data apa yang dapat ditolerir oleh sistem?
f. Apa pengaruh kombinasi tertentu dari data terhadap operasi sistem?