BAB II
LANDASAN TEORI

Teori-teori dasar yang terkait dalam pembuatan sistem informasi pada unit dagang Kencana Jaya, dijabarkan dalam sub-sub pokok bahasan di bawah ini.

2.1 Profil Toko Kencana Jaya

Kencana Jaya adalah sebuah usaha dagang yang menyediakan berbagai macam baut, mur dan tools. Usaha dagang yang dibangun dan dikelola oleh Dwi Atmoko ini berada dikawasan yang cukup mudah dijangkau. Sang pemilik yang mencoba untuk mengasah kemampuannya dalam bidang manajemen, pemilik mencoba membuka usaha ini dengan modal yang ia punya. Dengan bantuan keluarga dan teman-temannya, ia mampu membangun usaha ini dari awal dan teman-temannya pun membantu memasarkannya. Dengan berjalannya waktu, pemilik mampu memperluas pangsa pasarnya dan membuka anak cabang di area Bandar Lampung dan Natar. Kurang lebih selama sepuluh tahun sudah ia membuka uasaha ini, ia selalu meningkatkan kepuasan untuk para pelanggannya. Barang yang dijual dapat secara grosir dan ecer dengan harga yang tidak terlalu mahal. Pemilik pun selalu memperlengkap dagangannya sehingga para pelanggan yang datang tidak kecewa dan mendapatkan kepuasan. Secara idak langsung dengan adanya usaha ini pemilik telah menyediakan lapangan pekerjaan. Sampai saat ini kurang lebih sudah ada 12 orang yang mampu dipekerjakan. Tidak akan berhenti sampai disini karna sang pemilik akan terus memajukan usahanya ini agar terus berkembang lebih pesat lagi.

2.2 Sistem Informasi

Sesungguhnya yang dimaksud sistem informasi tidak harus melibatkan komputer. Sistem informasi yang menggunakan komputer biasa disebut sistem informasi

penting. Di buku ini, yang dimaksud dengan sistem informasi adalah sistem informasi berbasis komputer.

Ada beragam definisi sistem informasi, sebagaimana tercantum pada Tabel 2.1. Berdasarkan berbagai definisi tersebut, dapat disimpulkan bahwa sistem informasi mencangkup sejumlah komponen (manusia, komputer, teknologi informasi dan prosedur kerja), ada sesuatu yang diproses (data menjadi informasi) dan dimaksudkan untuk mencapai suatu sasaran atau tujuan (Kadir, 2014).

Tabel 2.1 Definisi Sistem Informasi

Sumber	Definisi
Alter (1992)	Sistem informasi adalah kombinasi antar prosedur kerja, informasi, orang dan teknologi informasi yang diorganisasikan untuk mencapai tujuan dalam sebuah organisasi.
Bodnar dan Hopwood (1993)	Sistem informasi adalah kumpulan perangkat keras dan peragkat lunak yang dirancang untuk mentransformasikan data ke dalam bentuk informasi yang berguna.
Gelinas, Oram dan Wiggins (1990)	Sistem informasi adalah suatu sistem buatan manusia yang secara umum terdiri atas sekumpulan komponen berbasis komputer dan manual yang dibuat untuk menghimpun, menyimpan dan mengelola data serta menyediakan informasi keluaran kepada para pemakai.
Hall (2001)	Sistem informasi adalah sebuah rangkaian prosedur formal, dimana data dikelompokkan, diproses menjadi informasi dan didistribusikan kepada para pemakai.
Turban, McLean dan Wetherbe	Sebuah sistem informasi mengumpulkan, memproses, mentimpan, menganalisis dan

(1999)	menyebarkan informasi untuk tujuam yang spesifik.
Wilkinson (1992)	Sistem informasi adalah kerangka kerja yang mengoordinasikan sumber daya (manusia dan komputer) untuk mengubah masukan (*input*) menjadi keluaran (informasi) guna mencapai sasaran-sasaran perusahaan.

2.3 Pemograman Berorientasi Objek

Metodologi berorientasi objek adalah suatu strategi pembangunan perangkat lunak yang mengorganisasikan perangkat lunak sebagai komponen objek yang berisi data dan operasi yang diberlakukan terhadapnya (Rosa, 2011). Metodologi berorientasi objek merupakan suatu cara bagaimana sistem perangkat lunka dibanggun melalui pendekatan objek secara sistematis. Metodologi berorientasi objek didasarkan pada penerapan prinsip-prinsip pengelolaan kompleksitas, yang meliputi rangkaian aktivitas analisis berorientasi objek, perancangan berorientasi objek, pemograman berorientasi objek dan pengujian berorientasi objek. Keuntungan menggunakan metodologi pemograman berorientasi objek adalah sebagai berikut :

a. Meningkatkan produktivitas, karena kelas dan objek yang ditemukan dalam suatu masalah masih dapat dipakai ulang untuk masalah lainnya yang melibatkan objek tersebut (*reusable*).
b. Kecepatan pengembangan, karena sistem yang dibangun baik dan benar pada saat analisis dan perancangan akan menyebabkan berkurangnya kesalahan pada saat pengodean.
c. Kemudahan pemeliharaan, pola-pola yang cenderung tetap dan stabil dapat dipisahkan dan pola-pola yang mungkin sering berubah-ubah.
d. Adanya konsistensi, karena sifat pewarisan dan pengurangan notasi yang sama pada saat analisis, perancangan maupun pengodean.
e. Meningkatkan kualitas perangkat lunak, karena pendekatan pengembangan lebih dekat dengan dunia nyata dan adanya konsistensi pada saat

pengembangannya, perangkat lunak yang dihasilkan akan mampu memenuhi kebutuhan pemakai serta mempunyai sedikit kesalahan.

2.4 Konsep Dasar Berorientasi Objek

Dalam rekayasa perangkat lunak, konsep pendekatan berorientasi objek dapat diterapkan pada tahap analisis, perancangan, pemograman dan pengujian perangkat lunak (Rosa, 2011). Beberapa konsep dasar yang harus dipahami tentang metodologi berorientasi objek adalah sebagai berikut :

a. Kelas (*class*)

   Kelas adalah kumpulan objek-objek dengan karakteristik yang sama dan memiliki sifat (atribut). Secara teknis, kelas adalah sebuah struktur tertentu dalam pembuatan perangkat lunak. Kelas merupakan bentuk struktur pada kode program yang menggunakan metodologi berorientasi objek.

b. Objek (*object*)

   Objek adalah abstraksi dan sesuatu yang mewakili dunia nyata seperti benda, manusia, satuan organisasi, tempat, kejadian, struktur dan hal-hal lainnya yang bersifat abstrak. Objek merupakan suatu entitas yang mampu menyimpan informasi (status) dan mempunyai operasi (kelakuan) yang dapat berpengaruh pada status objeknya.

c. Metode (*method*)

   Operasi atau metode sebuah kelas hampir sama dengan fungsi atau prosedur pada metodologi struktural. Sebuah kelas boleh memiliki lebih dari satu metode atau operasi.

d. Atribut (*attribute*)

   Atribut dapat berupa nilai atau elemen-elemen data yang dimiliki oleh objek dalam kelas.

e. Abstraksi (*abstraction*)

   Prinsip untuk merepresentasikan dunia nyata yang kompleks menjadi satu bentuk model yang sederhana dengan mengabaikan aspek-aspek lain yang tidak sesuai dengan permasalahan.

f. Enkapulasi (*encapulation*)

Pembungkusan atribut dan layanan (operasi-operasi) yang dipunyai objek untuk menyembunyikan implementasi dan objek sehingga objek lain tidak mengetahui cara kerjanya.

g. Pewarisan (*inheritance*)

Mekasnisme yang memungkinkan satu objek mewarisi sebagian atau seluruh definisi dan objek lain sebagai bagian dari dirinya.

h. Antarmuka (*interface*)

Antarmuka sangat mirip dengan kelas, akan tetapi tanpa atribut kelas dan memiliki metode yang dideklarasikan tanpa isi. Deklarasi metode pada sebuah *interface* dapat diimplementasikan oleh kelas lain.

i. Reusability

Pemanfaatan kembali objek yang sudah didefinisikan untuk suatu permasalahan pada permasalahan lainnya yang melibatkan objek tersebut.

j. Generalisasi dan Spealisasi

Menunjukkan hubungan antara kelas dan objek yang umum dengan kelas dan objek yang khusus.

k. Komunikasi Antar Objek

Komunikasi antar objek dilakukan lewat pesan yang dikirim dari satu objek ke objek lainnya.

l. Poliformisme (*polymorphism*)

Kemampuan suatu objek untuk digunakan dibanyak tujuan yang berbeda dengan nama yang sama sehingga menghemat baris program.

m. Package

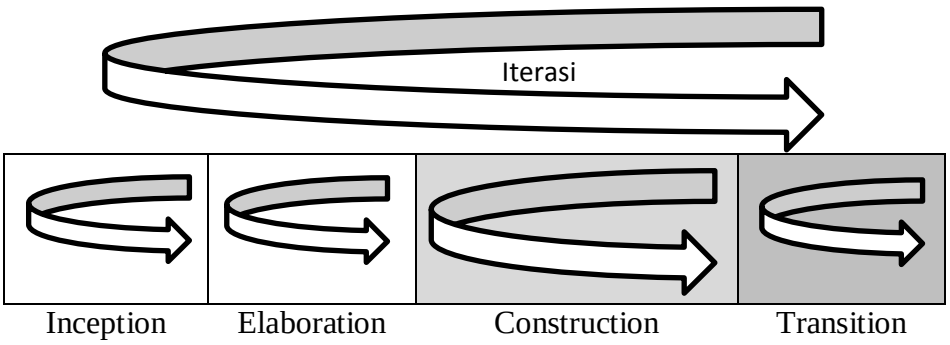
Merupakan sebuah kontainer atau kemasan yang dapat digunakan untuk mengelompokkan kelas-kelas sehingga memungkinkan beberapa kelas yang bernama sama disimpan dalam *package* yang bebeda.

2.5 Metode Pengembangan Perangkat Lunak

Unified Process atau dikenal juga dengan proses iteratif dan inkremental merupakan sebuah proses pengembangan perangkat lunak yang dilakukan secara

iteratif (berulang) dan inkremental (bertahap dengan proses menaik). Iteratif bisa dilakukan di dalam setiap tahap atau iteratif tahap pada proses pengembangan perangkat lunak untuk menghasilkan perbaikan fungsi yang inkremental, dimana setiap iterasi akan memperbaiki iterasi berikutnya (Rosa, 2011). Salah satu *Unified Process* yang terkenal adalah RUP (*Rational Unified Process*).

RUP adalah pendekatan pengembangan perangkat lunak yang dilakukan berulang-ulang , fokus pada arsitektur, lebih diarahkan berdasarkan penggunaan kasus (*use case driven*). RUP merupakan proses rekayasa perangkat lunak dengan pendefinisian yang baik dan penstrukturan yang baik. RUP memiliki empat buah tahap fase, yaitu seperti pada Gambar 2.1.



Gambar 2.1 Alur Hidup RUP

a. *Inception* (permulaan)

Tahap ini leih pada memodelkan bisnis yang dibutuhkan dan mendefinisikan kebutuhan akan sistem yang akan dibuat. Tahap yang dibutuhkan pada permulaan ini adalah :

1. Memahami ruang lingkup dari proyek (termasuk biaya, waktu, kebutuhan, resiko dan lainnya).
2. Membangun kasus bisnis yang dibutuhkan.

Hasil yang diharapkan pada tahap ini adalah memenuhi *lifecycle objective milestobe* (batas/tonggak objektif dari siklus) dengan kriteria berikut :

1. Umpan balik dari pendefinisian ruang lingkup, perkiraan biaya dan perkikraan jadwal.
2. Kebutuhan dimengerti dengan pasti dan sejalan dengan kasus primer yang dibutuhkan.
3. Kredibilitas dari perkiraan biaya, perkiraan jadwal, penentuan sekala prioritas, risiko dan proses pengembangan.
4. Ruang lingkup purwarupa (prototype) yang akan dikembangkan.
5. Membangun garis dasar dengan membandingkan perencanaan aktual dengan perencanaan yang direncanakan.

Jika pada akhir tahap ini target yang diinginkan tidak dicapai maka dapat dibatalkan atau diulang kembali setelah dirancang ulang agar kriteria yang diinginkan dapat dicapai.

b. *Elaboration* (perluasan atau perencanaan)

Tahap ini lebih difkuskan pada perencanaan arsitektur sistem. tahap ini lebih pada analisis dan desain sistem serta implementasi sistem yang fokus pada purwarupa sistem (*prototype*). Hasil yang diharapkan pada tahap ini adalah memenuhi *lifecycle objective milestobe* (batas/tonggak objektif dari siklus) dengan kriteria berikut :

1. Model kasus yang digunakan (*use case*) dimana kasus dan aktor yang terlihat telah didefinisikan dan sebagian besar kasus harus dikembangkan.
2. Deskripsi dari arsitektur perangkat lunak telah dibuat.
3. Rancangan aarsitektur yang dapat diimplementasikan dan mengimplementasikan *use case*.
4. Kasus bisnis atau proses bisnis dan daftar resiko yang sudah mengalami perbaikan.
5. Rencana pengembangan untuk seluruh proyek telah dibuat.
6. Purwarupa (*prototype*) yang dapat didemonstrasikan untuk mengurangi setiap resiko teknis yang diidentifikasi.

Jika pada akhir tahap ini target yang diinginkan tidak dicapai, maka dapat dibatalkan atau diulang kembali.

c. *Construction* (konstruksi)

Tahap ini fokus pada pengembangan komponen dan fitur-fitur sistem. tahap ini lebih pada implementasi dan pengujian sistem yang fokus pada implementasi perangkat lunak atau kode program. Tahap ini menghasilkan produk perangkat lunak dimana menjadi syarat dari *Initial Operational Capability Milestone* atau batas/tonggak kemampuan operasional awal.

d. *Transition* (transisi)

Tahap ini lebih pada *deployment* atau inisialisasi sistem agar dapat dimengerti *oleh user*. Tahap ini menghasilkan prosuk perangkat lunak dimana menjadi syarat dari *Initial Operational Capability Milestone* atau batas/tonggak kemampuan operasional awal. Aktivitas pada tahap ini termasuk pada pelatihan *user*, pemeliharaan dan pengujian sistem.

2.6 Konsep Basis Data

Basis data (*database*) adalah suatu pengorganisasian sekumpulan data yang saling terkait sehingga memudahkan aktivitas untuk memperoleh informasi. Basis data dimaksudkan untuk mengatasi problem pada sistem yang memakai pendekatan berbasis berkas.

Untuk mengelola basis data diperlukan perangkat lunak yang disebut *Database Management System* (DBMS). DBMS adalah perangkat lunak sistem yang memungkinkan para pemakai membuat, memelihara, mengontrol, dan mengakses basis data dengan cara yang praktis dan efisien. DBMS dapat digunakan untuk mengakomodasikan berbagai macam pemakai yang memiliki kebutuhan akses yang berbeda-beda (Kadir, 2014).

Suatu aplikasi dapat berkomunikasi melalui DBMS untuk mengakses basis data dan kemudian membetuk laporan-laporan Selain itu, pemakai juga bisa berinyeraksi secara langsung dengan DBMS untuk mengakses basis data, baik

untuk keperluan meminta ataupun untuk melakukan perubahan data. Interaksi secara langsung dengan basis data memungkinkan pemakai untuk memperoleh informasi-informasi yang dibutuhkan sewaktu-waktu dan bersifat sementara, tanpa memerlukan bantuan pemrogram. Umumnya DBMS menyediakan fitur-fitur sebagai berikut :

a. Independensi Data Program

   Karena basia data ditangani oleh DBMS, program dapat ditulis sehingga tidak tergantung pada stuktur data dalam basis data. Dengan perkataan lain, program tidak akan terpengaruh sekiranya bentuk fisik data diubah.

b. Keamanan

   Keamanan dimaksudkan untuk mencegah pengaksesan data oleh orang yang tidak berwewenang.

c. Integritas

   Hal ini ditujukan untuk menjaga agar data selalu dalam keadaan yang valid dan konsisten.

d. Konkurensi

   Konkurensi memungkinkandata dapat diakses oleh banyak pemakai tanpa menimbulkan masalah.

e. Pemulihan (*Recovery*)

   DBMS menyediakan mekanisme untuk mengembalikan basis data ke keadaan semula yang konsisten sekiranya terjadi gangguan perangkat keras atau kegagalan perangkat lunak.

f. Katalog Sistem

   Katalog sistem adalah deskripsi tentang data yang terkandung dalam basis data yang dapat diakses oleh pemakai.

g. Perangkat Produktivitas

   Untuk menyediakan kemudahan bagl pemakai dan meningkatkan produktivitas, DBMS menyediakan sejumlah perangkakat produktivitas seperti pembangkit *query* dan pembangkit laporan.

Komponen-komponen yang menyusun lingkungan DBMS terdiri atas:

a. Perankat Keras

   Perangkat keras digunakan untuk menjalankan DBMS beserta aplikasi-aplikasinya. Perangkat keras berupa komputer dan periferal pendukungnya. Komputer dapat berupa *PC, mini komputer, main frame* dan lain-lain.

b. Perangkat Lunak

   Komponen perangkat lunak mencakup DBMS itu sendiri, program aplikasi serta perangkat lunak pendukung untuk komputer dan jaringan. Program aplikasi dapat dibangun dengan menggunakan bahasa pemrograman seperti *C++, Pascal, Delphi atau Visual BASIC*.

c. Data

   Bagi sisi pemakai, komponen terpenting dalam DBMS adalah data karena dari data inilah pemakai dapat memperoleh informasi yang sesuai dengan kebutuhan masing-masing.

d. Prosedur

   Prosedur adalah petujuk tertulis yang berisi cara merancang hingga menggunakan basis data. Beberapa hal yang dimasukkan dalam prosedur :

   1. Cara masuk ke DBMS (*login*).
   2. Cara memekai fasilitas-fasilitas tertentu dalam DBMS maupun cara menggunakan aplikasi.
   3. Cara mengaktifkan dan menhentikan DBMS.
   4. Cara membuat cadangan basis data dan cara mengembalikan cadangan ke DBMS.

e. Orang

   Komponen orang dapat dibagi menjadi tiga kelompok, yaitu :

   1. Pemakai akhir (*end-user*).
   2. Pemogram aplikasi.
   3. Administrator basis data.

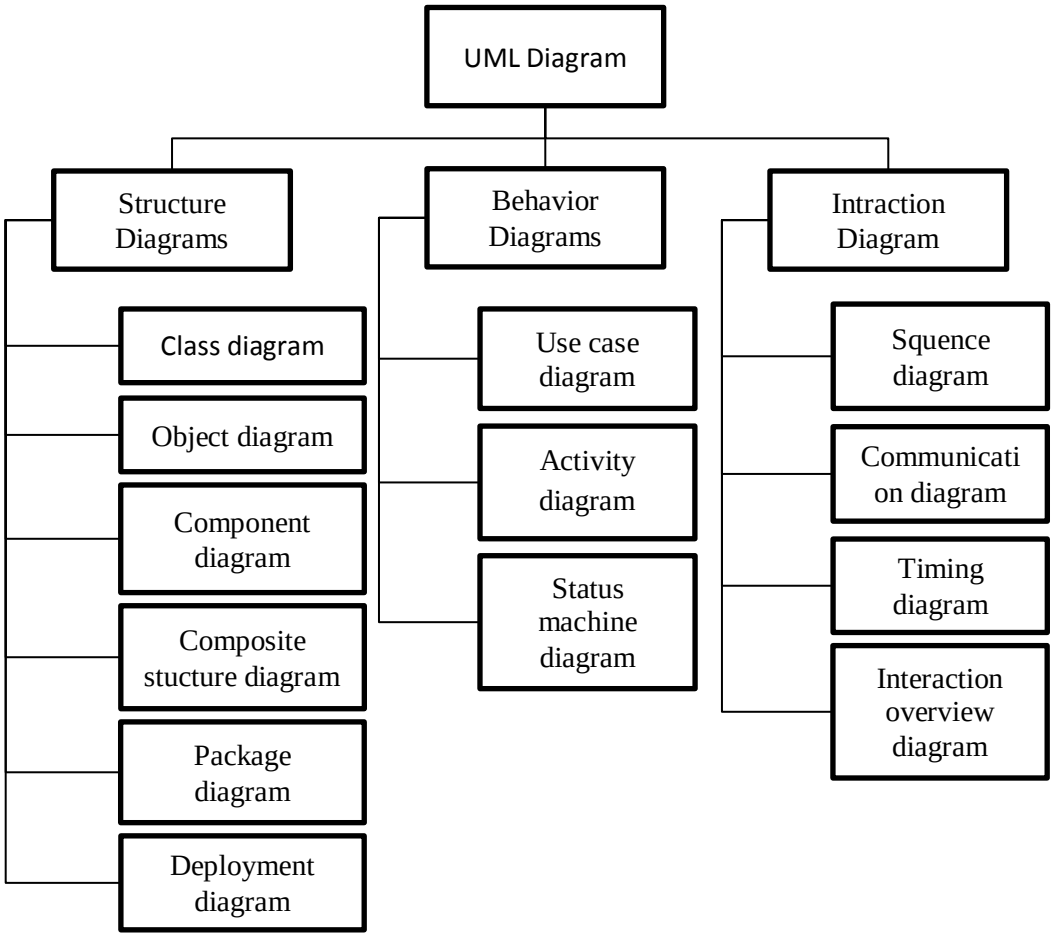
2.7 Alat Bantu Perancangan Sistem

Alat bantu perancangan sistem yang digunakan dalam merancang dan membangun sistem informasi unit dagang Kencana Jaya adalah UML (*use case diagram, activity diagram*) dan struktur *database*.

2.7.1 UML (*Unified Modeling Language*)

Banyak orang yang telah membuat bahasa pemodelan pembangunan perangkat lunak yang sesuai dengan teknologi pemograman yang berkembang pada saat itu, misalnya yang sempat berkembang dan diguakan oleh banyak pihak adalah *Data Flow Diagram* (DFD) untuk memodelkan perangkat lunak yang menggunakan pemograman prosedural ataustruktural, kemudian juga ada *State Transition Diagram* (STD) yang digunakan untuk memodelkan sistem *real time* (wakyu nyata).

Pada perkembangan teknik pemograman berorientasi objek, munculah sebuah standarisasi bahasa pemodelan untuk pembangunan perangkat lunak yang dibangun dengan menggunakan teknik pemograman berorientasi objek, yaitu *Unified Modeling Language* (UML). UML muncul karena adanya kebutuhan pemodelan visual untuk menspesifikasikan, menggambarkan, membangun dan mendokumentasi dari sistem perangkat lunak. UML terdiri dari 13 macam diagram yang dikelompokkan dalam tiga kategori, yaitu seperti pada Gambar 2.2 (Rosa, 2011).



Gambar 2.2 Diagram UML

Penjelasan dari pembagian kategori tersebut adalah :

a. *Structure diagram*, yaitu kumpulan diagram yang digunakan untuk menggambarkan suatu struktur statis dari sistem yang dimodelkan.
b. *Behavior diagram*, yaitu kumpulan diagram yang digunakan untuk menggambarkan kelakuan sistem atau rangkaian perubahan yang terjadi pada sebuah sistem.
c. *Interaction diagram*, yaitu kumpulan diagram yang digunakan untuk menggambarkan interaksi sistem dengan sistem lain maupun interaksi antar sub sistem pada suatu sistem.

2.7.1.1 Use Case

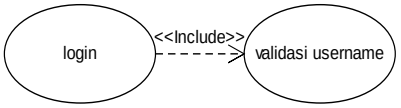
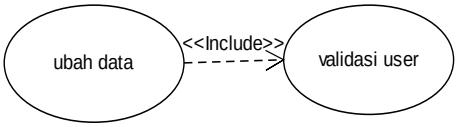
Use case mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. Secara kasar, *use case* digunakan untuk mengetahui fungsi apa saja yang ada di dalam ebuah sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi itu. Syarat penamaan pada *use case* adalah nama didefinisikan sesimpel mungkin dan dapat dipahami (Rosa, 2011). Ada dua hal utama pada *use case* yaitu pendefinisian apa yang dibuat aktor dan *use case*.

a. Aktor merupakan orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat di luar sistem informasi yang akan dibuat itu sendiri, jadi, walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang.
b. *Use case* merupakan fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor.

Tabel 2.2 Simbol *Use Case* Diagram

Keterangan	Simbol	Deskripsi
Use Case		Fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor; biasanya dinyatakan dengan menggunakan kata kerja diawal-awal frase nama *use case*
Aktor		Orang, proses atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat di luar itu sendiri. Jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang; biasanya dinyatakan menggunakan kata benda diawal frase nama aktor.

Asosiasi	———	Komunikasi antara aktor dan *use case* yang berpartisipasi pada *use case* atau *use case* memiliki interaksi dengan aktor.
Ekstensi	«extends» ——▷	Relasi use case tambahan ke sebuah *use case*, dimana *use case* yang ditambahkan dapat berdiri sendiri walau tanpa *use case* tambahan itu; mirip dengan prinsip *inheritance* pada pemograman berorientasi objek; biasanya *use case* tambahan memiliki nama depan yang sama dengan use case yang ditambahkan, misal  Arah panah mengarah pada *use case* yang ditambahkan.
Generalisasi	——→	Hubungan generalisasi dan spesialisasi (umum-khusus) antara dua buah *use case* dimana fungsi yang satu adalah fungsi yang lebih umum dari lainnya, misalnya :  Arah panah mengarah pada *use case* yang menjadi generalisasinya (umum).
Menggunakan/*include/uses*	<<Include>> - - - - - - ▷ «uses» ——▷	Ada dua sudut pandang yang cukup besar mengenai *include* di *use case* : a. Include berarti use case yang ditambahkan akan selalu dipanggil

		saat use case tambahan dijalankan, misal pada kasus berikut :  b. Include berarti use case yang tambahan akan selalu melakukan pengecekan apakah use case yang ditambahkan telah dijalankan sebelum use case tambahan dijalankan, misal pada kasus berikut :  Ke dua interpretasi di atas dapat dianut salah satu atau keduanya tergantung pada pertimbangan dan interpretasi yang dibutuhkan.

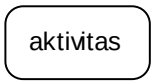
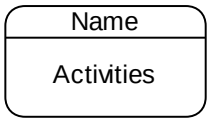
2.7.1.2 Activity Diagram

Diagram aktivitas atau *activity diagram* menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis. Yang perlu diperhatikan disini adalah bahwa diagram aktivitas menggambarkan aktivitas sistem bukan apa yang dilakukan aktor, jadi aktivitas yang dapta dilakukan oleh sistem. Diagram aktivitas juga banyak digunakan untuk mendefinisikan hal-hal berikut :

a. Rancangan proses bisnis dimana setiap urutan aktivitas yang digambarkan merupakan proses bisnis sistem yang didefinisikan.
b. Urutan atau pengelompokan tampilan dari sistem/*user interface* dimana setiap aktivitas dianggap memiliki sebuah rancangan antarmuka tampilan.

c. Rancangan pengujian dimana setiap aktivitas dianggap memerlukan sebuah pengujian yang perlu didefinisikan kasus ujinya.

Tabel 2.3 Simbol Diagram Aktivitas

Keterangan	Simbol	Deskripsi
Status awal		Status awal aktivitas sistem, sebuah diagram aktivitas memiliki sebuah status awal.
Aktivitas		Aktivitas yang dilakukan sistem, aktivitas biasanya diawali dengan kata kerja.
Percabangan		Asosiasi percabangan dimana jika ada pilihan aktivitas lebih dari satu.
Penggabungan		Asosiasi penggabungan dimana lebih dari satu aktivitas digabungkan menjadi satu.
Swimlane		Memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi.
Status akhir		Status akhir yang dilakukan sistem, sebuah diagram aktivitas memiliki sebuah status akhir.

2.7.2 Struktur *Database*

Database adalah kumpulan *file* yang saling berelasi, relasi tersebut biasa ditunjukkan dengan kunci dari tiap *file* yang ada. Satu basis menunjukkan satu kumpulan data yang dipakai dalam satu lingkup perusahaan atau instansi. (Rosa, 2011). Untuk membentuk suatu *database*, diperlukan jenjang data yang dimulai dari :

a. *Character*

   Bagian terkecil dapat berupa angka, huruf ataupun karakter khusus yang membentuk suatu item data.

b. *Field-Field*

   Kumpulan dari karakter-karakter suatu field menggunakan suatu atribut dari *record* menunjukkan suatu item dari data.

c. *Record*, kumpulan dari *field-field*.

d. *File*

   Kumpulan dari item data yang diatur dalam suatu *record* dimana item-item data tersebut dimanipulasi untuk proses tertentu.

e. Kamus Data

   Model yang bertujuan membantu pelaku sistem untuk dapat mengerti aplikasi secara detail dan mengorganisasi semua elemen aplikasi data yang digunakan dalam sistem sehingga pemakai dan penganalisa sistem mempunyai dasar pengertian yang sama tentang masukan, keluaran, penyimpanan dan proses

2.8 Kebutuhan Perangkat Lunak

Perangkat lunak yang digunakan untuk membantu dalam proses pembuatan sistem informasi usaha dagang Kencana Jaya adalah *.Net Framework, VB.Net* dan *MySQL*.

2.8.1 .Net Framework

.Net (dibaca : "dot net") *Framework* adalah *software* yang mendukung proses pengembangan dan eksekusi program dilingkungan *windows. Framework* tersebut dikembangkan oleh *Microsoft* (Budi, 2016). Terdapat dua bagian atau entitas penting di dalam *.Net Framework* yang perlu diketahui sebelum melakukan pembuatan program-program menggunakan f*ramework* ini, yaitu : *Class Library* dan *Common Language Runtime (CLR).*

a. *.Net Class Library*

.net Framework berisi sekumpulan *library* (pustaka) berupa kelas yang diatur dan dikelompokkan ke dalam bentuk hirarki *namespace.* Daftar kelas yang paling banyak digunakan disimpan di dalam *namespace system.** atau *microsoft.**. Kelas-kelas tersebut mengimplementasikan fungsi-fungsi umum yang paling digunakan dalam pengembangan program, seperti baca/tulis *file,* interaksi dengan *database,* manipulasi dokumen XML dan sebagainya. *Class Library* dalam *.Net Framework* itu sendiri diagi ke dalam dua kelompok, yaitu *Base Class Library* dan *Framework Class Library.*

Base Class Library (BCL) berisi kumpulan kelas inti *(subnet* dari seluruh kelas yang terdapat di dalam *.Net Class Library*) yang menyediakan fungsi-fungsi dasar dari CLR. Kelas-kelas yang tersimpan dalam *file micorlib.dll* serta berupa kelas yang terdapat di dalam *file System.dll* dan *System.core.dll* ditetapkan sebagai bagian dari BCL.

Framework Class Library (FCL) merupakan *superset* dari BCL dan mengacu ke seluruh *library* yang terdapat dalam *.Net Class Library*. Dengan demikian, FCL sebenarnya merupakan nama atau istilah lain dari *.Net Class Library*. FCL berisi sekumpulan kelas untuk pembuatan program yang melibatkan *Windown Form* (WF), *ADO.Net*, *ASP.Net*, *Language Intregated Query* (LINQ), *Windows Presentation Fundation* (WCF) dan lain-lain.

b. Common Language Runtime (CLR)

Common Language Runtime adalah lingkungan atau sistem *(virtual machine)* yang mengatur proses eksekusi dari program-program yang ditulis menggunakan library.Net. pada saat melakukan kompilasi kode program yang ditulis menggunakan *C#, VB.Net. VC.Net,* maupun *J#,* hasil yang diberikan oleh kompilator (*compiler*) sebenarnya bukan berupa *executable file (.exe),* melainkan *file* berisi kode khusus yang disebut *Microsoft Intermediate Language (MSIL).* File MSIL merupakan sekumpulan intruksi yang bersifat portable, yang dapat dijalankan di dalam semua jenis CPU yang sudah dipasang *.Net Framework*. Pada saat program dijalankan, CLR akan mengaktifkan kompilator JIT (*Just In-Time*) untuk mengubah *file* MSIL menjadi *file .exe*. Dengan demikian, yang dijalankan adalah *file .exe*

meskipun sebelumnya kode program akan diproses ke dalam file MSIL terlebih dahulu.

2.8.2 VB.Net

VB.NET adalah salah satu bahasa pemrograman komputer tingkat tinggi. Bahasa Pemrograman adalah perintah-perintah yang dimengerti oleh komputer untuk melakukan tugas-tugas tertentu. Bahasa pemrograman VB.NET dikembangkan oleh Microsoft, merupakan salah satu bahasa pemrograman yang *Object Oriented Program* (OOP) atau pemrograman yang berorientasi pada objek. Kata "Visual" menunjukkan cara yang digunakan untuk membuat *Graphical User Interface* (GUI) (Wawan, 2008).

2.8.2.1 Perbedaan VB dan *VB.Net*

Visual Basic (VB) adalah suatu development tool (alat/software untuk mengembangkan aplikasi) yang menggunakan bahasa pemograman BASIC sebagai bahasa pengontrolnya. Bahasa BASIC adalah bahasa pemograman yang sempat populer di era 80-an. *Visual Basic* adalah bahasa pemograman *event-drivent* yang dikategorikan sebagai *software Rapid Application Development* (RAD). Berikut ini adalah sejarah rilis dari *Visual Basic* (Budi, 2016) :

a. *Visual Basic 1.0* (tahun 1991)
b. *Visual Basic 2.0* (tahun 1992)
c. *Visual Basic 3.0* (tahun 1993)
d. *Visual Basic 4.0* (tahun 1995)
e. *Visual Basic 5.0* (tahun 1997)
f. *Visual Basic 6.0* (tahun 1998)

Dengan menggunkaan *Visual Basic* (versi 1.0 sampai 6.0), para *programmer* dapat membuat aplikasi visual yang memanfaatkan komponen-komponen kontrol maupun *library* yang disediakan oleh *Visual Basic* itu sendiri. Dengan

kata lain, para *programmer* tidak dapat menggunakan *Visual Basic 6*.0 (atau sebelumnya) untuk mengkases *library* yang disediakan oleh .*Net Framework.*
Di akhir era 90-an, *Microsoft* mengembangkan .*Net Framework* yang awalnya diinialisasi dengan nama *Next Generation Windows Services* (NGWS). Pada tahun 2000, *Microsoft* merilis versi beta dari .Net 1.0. sejak adanya .*Net Framework*, diawali dari tahun 2002, *Microsoft* mulai mengembangkan *software* dengan label *Visual Studio.Net*, yang merupakan paket atau gabungan dari *software Visual Basic.Net, Visual C++ .Net* dan *Visual C#*. Dengan *Visual Studio .Net*, dalam mengembangkan aplikasi para *programmer Visual Basic* harus menggunakan *library* yang disediakan oleh .*Net Framework*, tidak menggunakan *library* bawaan dari *Visual Basic* itu sendiri (seperti yang diimplementasikan oleh VB 1.0 sampai VB 6.0)

2.8.3 MySQL

MySQL adalah salah satu jenis *database server* yang terkenal dan banyak digunakan untuk membangun aplikasi web yang menggunakan *database* sebagai sumber dan pengelolaan data. Kepopuleran *MySQL* antara lain karena *MySQL* menggunakan *SQL* sebagai bahasa dasarnya untuk mengakses *database*nya sehingga mudah untuk digunakan.

MySQL termasuk RDBMS (*Relational Database Management System*). Pada *MySQL*, sebuah d*atabase* mengandung satu atau sejumlah tabel. Tabel terdiri atas sejumlah kolom dan baris, dimana setiap kolom berisi sekumpulan data, dan baris merupakan sekumpulan data yang saling berkaitan dan membentuk informasi. Kolom biasanya disebut sebagai field dan informasi yang tersimpan dalam setiap baris disebut *record* (Rudyanto, 2011).

2.9 Penelitian Terkait

Penelitian yang terkait dengan penelitian yang sudah dilakukan sebelumnya adalah sebagai berikut :

a. Menurut Rocky, dalam penelitiannya menyimpulkan bahwa pengolahan data keluar masuk barang secara manual cukup memakan banyak waktu sebab mendata dan membuat laporannya pada lembar laporan tersendiri, selain itu buku arsip yang tidak sedikit akan memakan ruang penyimpanan data. Dengan komputerisasi sistem persediaan keluar masuk barang nantinya dapat memberikan informasi yang lebih baik dan efesien serta dapat membantu dalam pengawasan keluar masuk barang di Inside Distro.
b. Menurut Ika, dalam penelitiannya yang menyimpulkan bahwa dengan adanya pembuatan sistem informasi penjualan pada toko sehat jaya elektronik pacitan dari sistem yang masih konvensional menjadi sistem yang terkomputerisasi dapat digunakan dan dimengerti sesuai yang diharapkan terhadap pimpinan toko sehat jaya elektronik pacitan dan adanya sistem informasi penjualan pada toko sehat jaya elektronik pacitan yang terkomputerisasi, semua data dokumen penjualan yang sebelumnya masih konvensional sekarang menjadi tersimpan dengan baik.
c. Menurut Daniel, dalam penelitiannya menyimpulkan bahwa program sistem informasi penjualan yang direncanakan, penyajian laporan pembelian dan penjualan yang mudah serta menghasilkan informasi yang lebih cepat dan akurat sehingga memudahkan pemilik toko dalam mengontrol persediaan *(inventory)* barang serta membantu dalam mengambil keputusan dalam pembelian barang dan kebijakan dalam pemberian diskon oleh pemilik toko. Dengan menggunakan program ini sebagai alat bantu dalam transaksi penjualan, terjadi perbedaan waktu sejak pembeli masuk area penjualan sampai dengan keluar dari area penjualan terjadi efisiensi/pengurangan waktu sekian 63.68 % dari waktu rata-rata keseluruhan 10 jenis barang secara manual dimana waktu yang dibutuhkan 22.27 menit dengan menggunakan program menjadi 8.09 menit dan mengurangi tingkat kesalahan karena perhitungan telah dilakukan secara otomatis oleh program.
d. Menurut Rini, dalam penelitiannya yang menyimpulkan bahwa sistem informasi penjualan barang mempunyai fasilitas data stok barang, data supplier, transaksi penjualan, pembelian, retur, backup, restore, laporan dan

nota. Sehingga sistem dapat membantu proses penjualan barang toko Sumber Urip.