BAB II
TINJAUAN PUSTAKA

2.1 *Mobile*

Mobile application juga disebut dengan *mobile apps*, yaitu istilah yang digunakan untuk mendeskripsikan aplikasi internet yang berjalan pada *smartphone* atau piranti *mobile* lainnya. Aplikasi *mobile* biasanya membantu para penggunanya untuk terkoneksi dengan layanan *internet* biasa diakses pada sebuah personal komputer [2].

Aplikasi *mobile* adalah aplikasi yang telah dirancang khusus untuk platform mobile (misalnya *iOS, android,* atau *windows mobile*). Dalam banyak kasus, aplikasi mobile memiliki user interface dengan mekanisme interaksi unik yang disediakan oleh platform mobile, interoperabilitas dengan sumber daya berbasis *web* yang menyediakan akses ke beragam informasi yang relevan dengan aplikasi, dan kemampuan pemrosesan lokal untuk pengumpulan, analisis, dan format informasi dengan cara yang paling cocok untuk *platform mobile*. Selain itu aplikasi *mobile* menyediakan kemampuan penyimpanan persisten dalam *platform* [3].

2.2 *Marketplace*

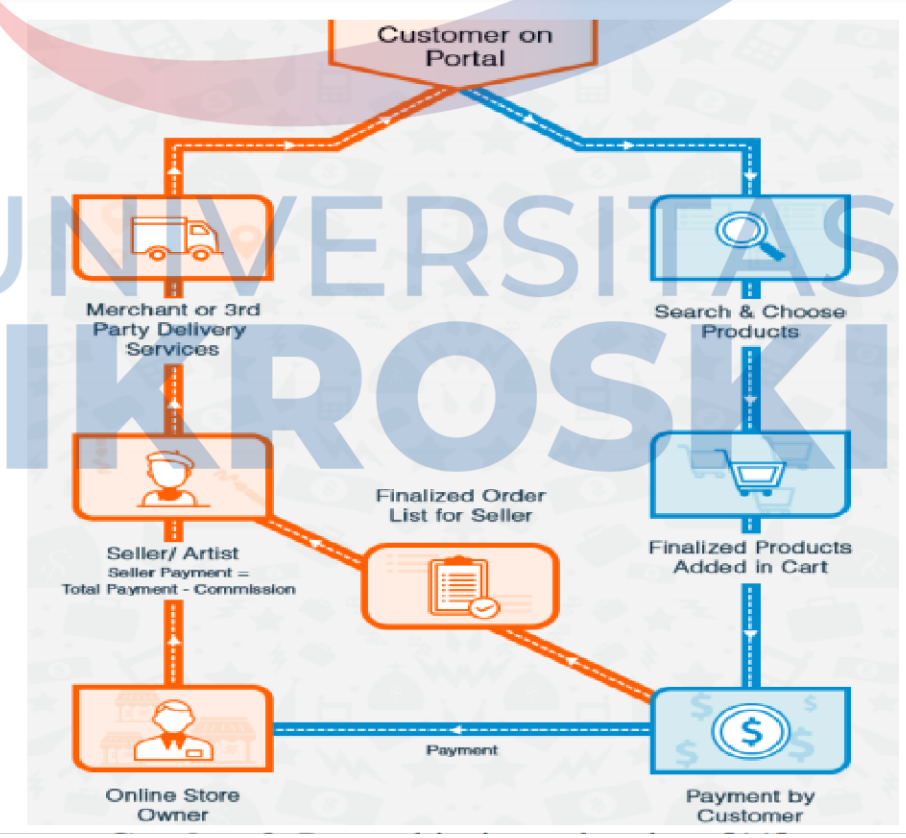
Marketplace adalah perantara antara penjual dan pembeli di dunia maya. Situs *marketplace* bertindak sebagai pihak ketiga dalam transaksi *online* deangn menyediakan tempat berjualan dan fasilitas pembayaran. Bisa dikatakan *marketplace* adalah *department store online* [4].

Marketplace merupakan model bisnis baru yang berkembang seiring pesatnya perkembangan infrastruktur teknologi informasi. *Marketplace* dirancang untuk meminimalisir proses bisnis yang kompleks sehingga tercipta efisiensi dan efektifitas, dengan adanya *Marketplace* setiap orang dapat melakukan aktivitas jual beli dengan mudah, cepat dan murah karena tidak ada batas ruang, jarak dan waktu. Indikator dari aktivitasnya *Marketplace* ditentukan oleh kemampuan *Marketplace* dalam menfasilitasi transaksi, mempertemukan penjual dan pembeli serta menyediakan infrastruktur. Sedangkan indikator efisiensi berkaitan dengan biaya yang diberikan *Marketplace*.

Marketplace merupakan *platform* transaksi bisnis *online* yang menyediakan metode untuk memfasilitasi transaksi komersil seperti menjual barang, jasa ataupun informasi secara online antara pembeli dan penjual [5].

2.2.1 Proses *Marketplace*

Proses *Marketplace* Sistem penjualan berbasis *marketplace* merupakan kegiatan transaksi bisnis yang dilakukan secara *online* dengan cukup praktis tanpa harus berkoban lebih dan cukup melakukan transaksi bisnis melaui hp, laptop, atau alat telekomunikasi lainnya. *Marketplace* juga dapat menunjukan banyaknya pengaruh yang berdampak positif baik bagi pengguna dalam mencari informasi maupun untuk kegiatan bisnis. *Marketplace* juga memiliki segmentasi penerapan yang luas secara garis besar, *marketplace* diterapkan untuk melaksanakan aktivitas ekonomi *business to business, business to customer,* dan *customer to customer* [5].



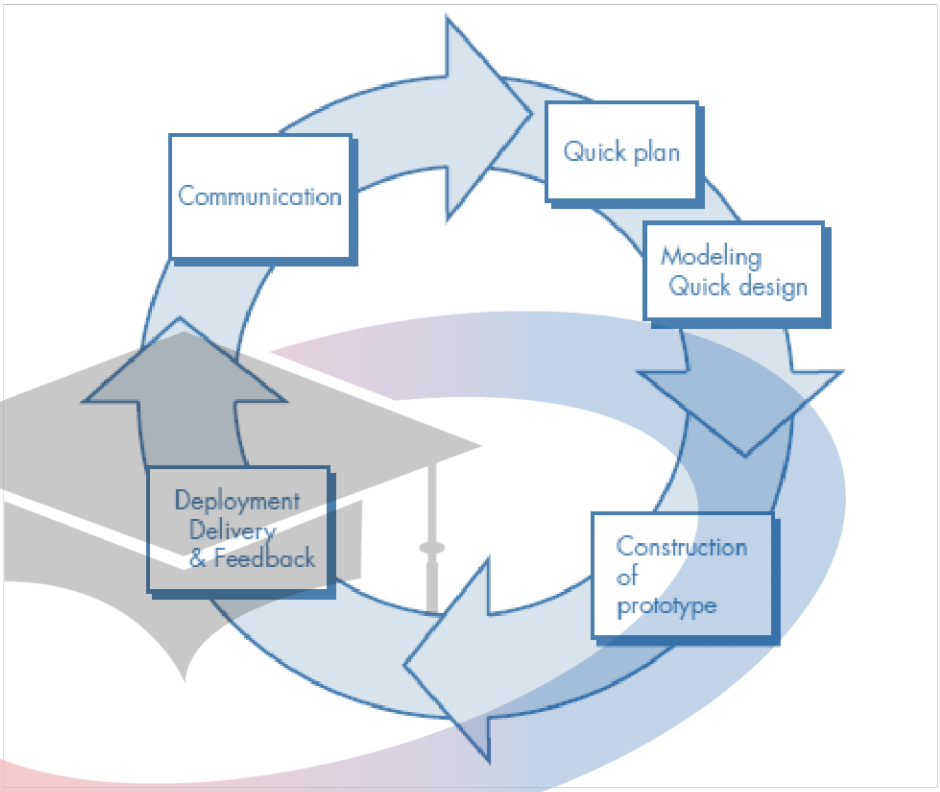
Gambar 2. 1. Proses Bisnis Marketplace

2.3 Metode *Prototyping*

Menurut literatur, yang dimaksud dengan *Prototype* adalah "model pertama", yang sering digunakan oleh perusahaan industri yang memproduksi barang secara masal. Tetapi dalam kaitannya dengan sistem informasi definisi kedua dari *Webster* yang menyebutkan bahwa *"prototype is an individual that exhibits the essential features 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 [6].

Prototype merupakan model kerja dari sebuah sistem informasi manajemen yang belum lengkap. Para pengembang sistem informasi, melakukan pertemuan-pertemuan intensif dengan user untuk menampung informasi yang akan dijadikan dasar dalam menyusun *Prototype* dari sistem informasi manajemen yang akan disajikan kelak. *Prototype* yang dihasilkan kemudian dipresentasikan kepada *user*, dan user diberikan kesempatan untuk memberi masukan-masukan sehingga sistem informasi manajemen yang dihasilkan betul-betul sesuai dengan keinginan dan kebutuhan user. Perubahan dan presentasi *Prototype* ini dapat dilakukan berkali - kali sampai dicapai kesepakatan bentuk sistem informasi manajemen yang akan diterapkan [6].

Proses *prototyping* (Gambar 2.2) dimulai dengan komunikasi. *User* bertemu dengan pemangku kepentingan lain untuk menentukan tujuan keseluruhan perangkat lunak, mengidentifikasi apapun persyaratan diketahui, dan garis besar area dimana definisi lebih lanjut wajib. Sebuah literasi *prototyping* direncanakan dengan cepat, dan pemodelan terjadi. Desain cepat berfokus pada representasi aspek-aspek perangkat lunak tersebut yang akan terlihat oleh pengguna akhir [7].



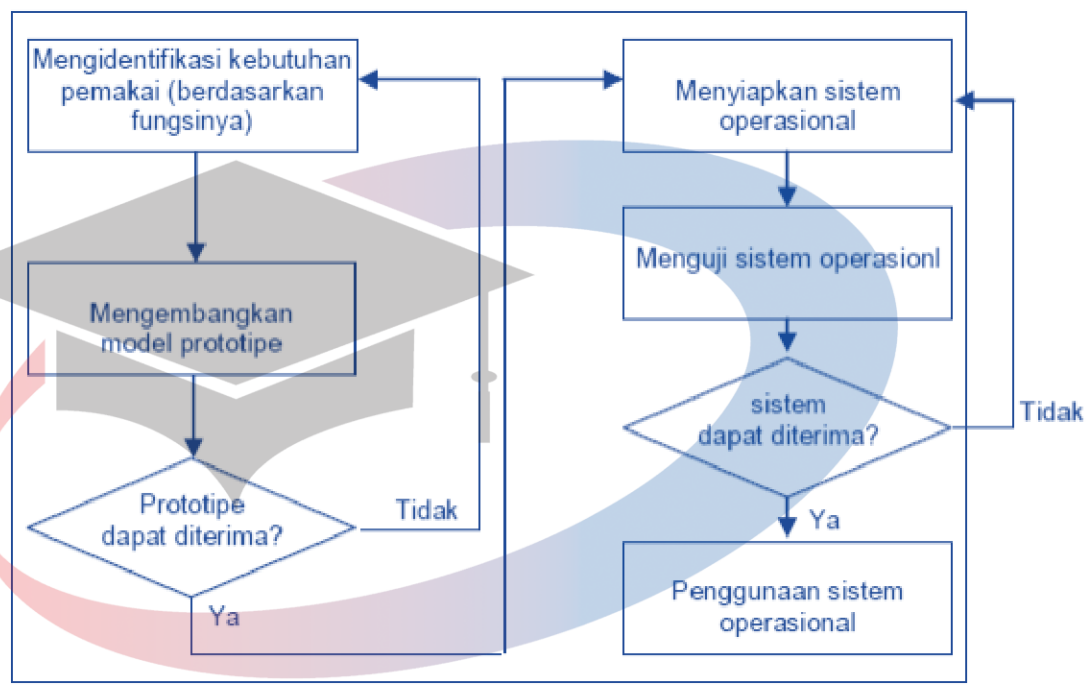
Gambar 2. 2. Proses Prototyping

Desain cepat mengarah pada pembangunan *Prototype*. *Prototype* dikerahkan dan dievaluasi oleh pemangku kepentingan, yang memberikan umpan balik yang digunakan untuk lebih lanjut perbaiki persyaratan. Iterasi terjadi ketika *Prototype* disetel untuk memenuhi kebutuhan berbagai pemangku kepentingan, sementara pada saat yang sama memungkinkan untuk lebih memahami apa yang perlu dilakukan. Idealnya, *Prototype* berfungsi sebagai mekanisme untuk mengidentifikasi persyaratan perangkat lunak [7].

2.3.1 Tahapan Dalam Metode *Prototyping*

Metode *Prototype* merupakan satu metode dalam perancangan, Sebagian *user* kesulitan mengungkapkan keinginannya untuk mendapatkan aplikasi yang sesuai dengan kebutuhannya. Kesulitan ini yang perlu diselesaikan oleh analis perancangan sistem dengan memahami kebutuhan user dan menerjemahkannya ke dalam bentuk model *(Prototype).* Dalam tahapan metode *prototyping* (Gambar 2.3) dimulai dengan mengidentifikasi kebutuhan pemakai berdasarkan fungsinya, mengembangkan model

prototyping, bila *prototyping* diterima dapat meyiapkan sistem operasional, kemudian melakukan pengujian sistem operasional jika penggunaan sistem diterima maka sistem operasional dapat digunakan.



Gambar 2. 3. Tahapan dalam Metode Prototyping

Berikut ini adalah beberapa tahapan metode *prototype* adalah sebagai berikut [6]:

1. Pengumpulan Kebutuhan

   Langkah pertama kali yang harus dilakukan dalam tahapan metode *prototype* adalah mengidentifikasi seluruh kebutuhan perangkat. Tahapan metode *prototype* yang sangat penting adalah analisis dan identifikasi kebutuhan garis besar dari sistem. selain itu akan diketahui langkah apa dan permasalahan yang akan dibuat dan dipecahkan. Pengumpulan kebutuhan sangat penting dalam proses ini.

2. Membangun *Prototype*

   Langkah selanjutnya dalam metode p*rototyping* adalah membangun *Prototype* yang berfokus pada penyajian pelanggan. Misalnya membuat input dan output hasil sistem. Sementara hanya *prototype* saja dulu selanjutnya akan ada tindak lanjut yang harus dikerjakan.

3. Evaluasi *Prototype*

   Sebelum melangkah kelangkah selanjutnya, ini bersifat wajib yaitu memeriksa langkah 1, dan karena ini adalah penentu keberhasilan dan proses yang sangat penting. Ketika langkah 1, dan 2 ada yang kurang atau salah kedepannya akan sulit sekali melanjutkan langkah selanjutnya.

4. Mengkodekan Sistem

   Sebelum pengkodean atau biasanya kita sebut proses *coding*, perlu kita ketahui terlebih dahulu pengkodean menggunakan bahasa pemrograman. Proses ini sangat sulit, karena mengaplikasikan kebutuhan dalam bentuk kode program.

5. Menguji Sistem

   Setelah pengkodean atau pengkodingan tentunya akan di *testing*. Banyak sekali cara untuk testing, misalkan menggunakan *white box* atau *black box.* Menggunakan *black box* menguji fungsi-fungsi tampilan apakah sudah benar dengan aplikasinya atau tidak.

6. Evaluasi *System*

   Mengevaluasi dari semua langkah yang pernah di lakukan. Sudah sesuai dengan kebutuhan atau belum. Jika belum atau masih ada revisi maka dapat mengulangi dan kembali di tahap 1 dan 2.

7. Menggunakan *System*

   Sistem sudah selesai dan siap di serahkan kepada pelanggan, dan jangan lupa untuk *maintenance* agar sistem terjaga dan berfungsi sebagaimana mestinya.

2.4 Konsep Dasar UML (*Unified Modeling Language*)

Unified Modeling Language adalah metodologi untuk mengembangkan sistem OOP dan seperangkat tool untuk mendukung pengembangan sistem. *Unified Modeling Language* didefinisikan sebagai bahasa pemodelan general purpose standar di bidang rekayasa perangkat lunak berorientasi objek. UML merupakan alat untuk menentukan dan digunakan untuk menentukan, memvisualisasikan, memodifikasi, membangun dan

mendokumentasikan artefak dari sistem perangkat lunak intensif yang berorientasi objek dalam pengembangan [8].

Adapun jenis – jenis diagram pada UML (*Unified Modeling Language*), yaitu [9]:

1. *Use Case Diagram*

   Use Case diagram merupakan pemodelan untuk sistem informasi yang akan dibuat. *Use Case* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi tersebut.

2. *Activity Diagram*

   Activity Diagram menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis.

3. *Sequence Diagram*

   Sequence Diagram menggambarkan kelakuan objek pada *Use Case* dengan mendeskripsikan waktu hidup objek dan pesan yang dikirimkan dan diterima antar objek

4. *Class Diagram*

   Merupakan hubungan antar kelas dan penjelasan detail tiap-tiap kelas di dalam model desain dari suatu sistem, juga memperlihatkan aturan-aturan dan tanggung jawab entitas yang menentukan perilaku sistem

Tujuan Pengguna UML (*Unified Modeling Language*) antara lain, yaitu [10]:

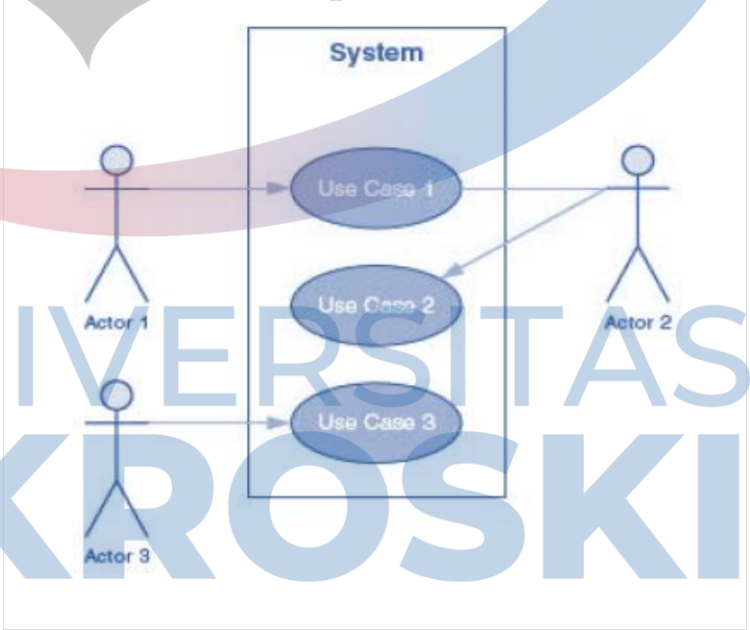
1. Memberikan bahasa pemodelan yang bebas dari berbagai bahasa pemrograman dan proses rekayasa.
2. Menyatukan praktek-praktek terbaik yang terdapat dalam pemodelan.
3. Memberikan model yang siap pakai, bahasa pemodelan visual yang ekspresif untuk mengembangkan dan saling menukar model dengan mudah dan dimengerti secara umum.

2.5 Teknik Pengembangan Sistem

2.5.1 *Use Case Diagram*

Use Case adalah rangkaian atau 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 dalam sebuah model serta direalisasikan oleh sebuah kolaborasi [10].

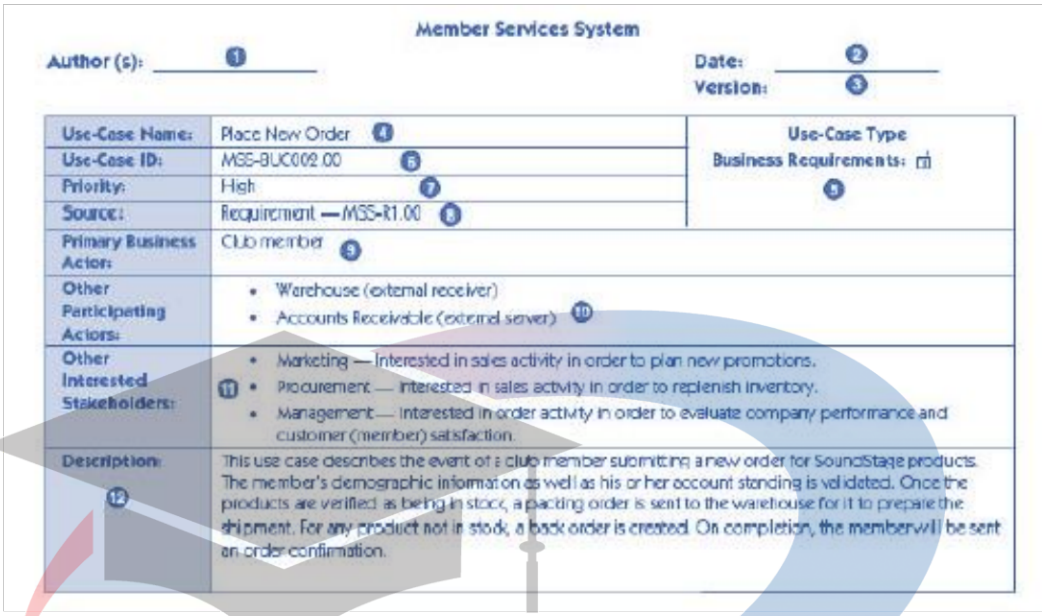
Use Case diagram yaitu diagram yang digunakan untuk menggambarkan hubungan antara sistem dengan aktor. Diagram ini hanya menggambarkan secara global. karena *Use Case* diagram hanya menggambarkan sistem secara global, maka elemen-elemen yang digunakan sangat sedikit, berikut ini elemen-elemen yang digunakan pada *Use Case diagram* [11]. Berikut adalah salah satu contoh *Use Case*:



Gambar 2. 4. Contoh Use Case

2.5.2 *Use Case Narrative*

Use Case Narrative yaitu uraian deskripsi dari *Use Case diagram* sehingga pengguna UML bisa mengetahui detail dari proses yang ada pada *Use Case diagram*. berikut ini elemen elemen yang digunakan pada *Use Case narrative* [11]. Contoh dari *Use Case narrative* dapat dilihat pada gambar 2.5 seperti berikut ini [11]:



Member Services System

Author (s): ❶ Date: ❷
Version: ❸

Use-Case Name:	Place New Order ❹	Use-Case Type Business Requirements: ☐ ❺
Use-Case ID:	MSS-BUC002.00 ❻	
Priority:	High ❼	
Source:	Requirement — MSS-R1.00 ❽	
Primary Business Actor:	Club member ❾	
Other Participating Actors:	• Warehouse (external receiver) • Accounts Receivable (external server) ❿	
Other Interested Stakeholders:	⓫ • Marketing — interested in sales activity in order to plan new promotions. • Procurement — interested in sales activity in order to replenish inventory. • Management — interested in order activity in order to evaluate company performance and customer (member) satisfaction.	
Description: ⓬	This use case describes the event of a club member submitting a new order for SoundStage products. The member's demographic information as well as his or her account standing is validated. Once the products are verified as being in stock, a packing order is sent to the warehouse for it to prepare the shipment. For any product not in stock, a back order is created. On completion, the member will be sent an order confirmation.	

Gambar 2. 5. Use Case narrative

Use Case narrative pada gambar 2. 5. akan dijelaskan dibawah ini yaitu [11]:

1. *Author* – Nama dari individu yang berkontribusi menulis *Use Case* dan yang menyediakan informasi pada *Use Case.*
2. *Date* – Tanggal terakhir *Use Case* dimodifikasi.
3. *Version* –Versi *Use Case* saat ini
4. *Use-case* name – Nama *Use Case* harus merepresentasikan tujuan dari *Use Case* tersebut. Penamaan harus dimulai dengan kata kerja
5. *Use-case type* – Dalam melakukan *use-case modeling*, *business requirements Use Case*, yang berfokus pada visi strategis dan tujuan dari berbagai *stakeholders*, dibuat pertama. Tipe *Use Case* ini berorientasi bisnis dan merefleksikan tampilan *high-level* dari sistem. *Use Case* ini bebas dari detail teknis dan dapat mengandung aktifitas manual dan juga aktifitas yang automatis. *Business requirements Use Case* menyediakan pemahaman umum tentang masalah domain dan cakupan tetapi tidak mengandung detil untuk berkomunikasi dengan pihak pengembang tentang apa yang sistem seharusnya bekerja.
6. *Use-case ID* – Merupakan identifier yang secara unik mengidentifikasi *Use Case.*

7. *Priority* – Menunjukan seberapa pentingnya *Use Case* yang dibuat. *Priority* ditunjukkan dengan tinggi, sedang atau rendah.
8. *Source* – Mendefinisikan *entity* yang memicu pembuatan dari *Use Case* ini. *Source* dapat berupa *requirements*, sesuatu dokumen spesifik atau sebuah *stakeholder*.
9. *Primary business actor* – Merupakan *stakeholder* yang sangat dipengaruhi ketika *Use Case* dieksekusi dengan menerima sesuatu yang dapat diukur dan dapat diobservasi.
10. *Other participating actors* – *Actors* lainnya yang berpartisipasi dalam *Use Case* untuk mencapai tujuan. *Actor* disini termasuk *initiating actors*, *facilitating actors, server/receiver actors* dan *secondary actors*. Selalu dijelaskan aksi dari *actors* tersebut.
11. *Interested stakeholder* – Seorang *stakeholder* adalah siapa saja yang terlibat dalam pembuatan dan pengoperasian sistem *software*. Seorang *interested stakeholder* adalah orang yangmemiliki kepentingan dari tujuan *Use Case*.
12. *Description* – Merupakan rangkuman singkat yang mengandung tujuan dari *Use Case* dan aktivitasnya.

2.5.3 *Activity Diagram*

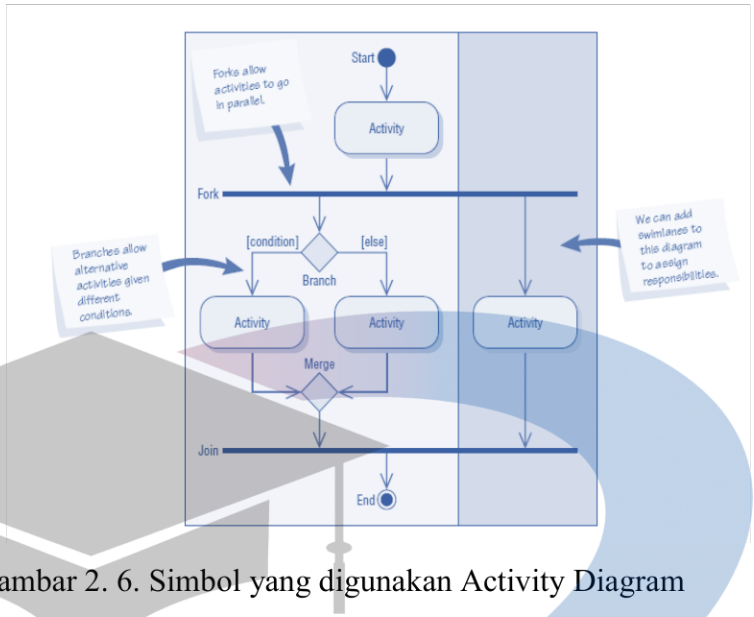
Activity diagram memodelkan *workflow* proses bisnis dan urutan aktivitas dalam sebuah proses. Diagram ini sangat mirip dengan *flowchart* karena memodelkan *workflow* dari suatu aktivitas ke status. *Activity diagram* juga bermanfaat untuk menggambarkan *parallel behaviour* atau menggambarkan interaksi antara beberapa *Use Case* [12].

Activity Diagram yaitu diagram yang digunakan untuk menggambarkan alur kerja (aktivitas) pada *Use Case* (proses), logika, proses dan hubungan antara aktor dengan alaur-alur kerja *Use Case* [11]. *Activity Diagram* terdiri dari beberapa simbol, yaitu sebagai berikut [13]:

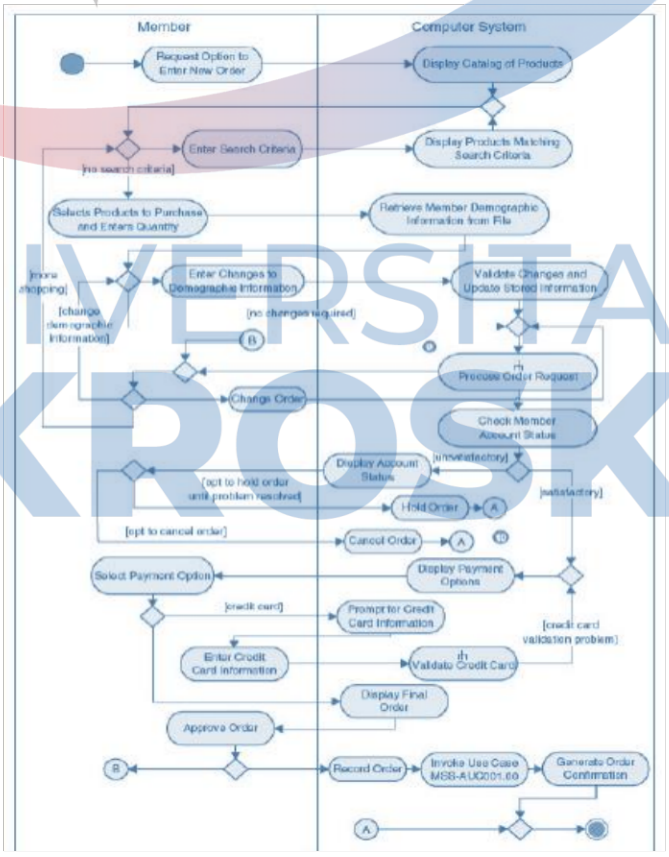
Tabel 2. 1. Simbol – Simbol pada Activity Diagram

No	Simbol	Keterangan	Definisi

1.		*Activity*	Memperlihatkan bagaimana masing-masing kelas antar muka Saling berinteraksi satu sama lain.
2.		*Action*	*State* dari sistem yang mencerminkan eksekusi dari suatu aksi.
3.		*Initial Node*	Bagaimana objek dibentuk atau diawali.
4.		*Activity final Node*	Bagaimana objek dibentuk dan dihancurkan.
5.		*Fork Node*	Satu aliran yang pada tahap tertentu berubah menjadi beberapa aliran
6.		*Swimlane*	Memisahkan organisasi bisnis yang bertanggung jawab terhadap aktifitas yang terjadi.
7.		*State Transtion*	Aliran dari aktifitas.
8.		*Decision*	Cabang keluaran dari *Condition d*apat lebih dari dua, tetapi biasanya sebagaian besar hanya berisi dua keluaran biner.



Gambar 2. 6. Simbol yang digunakan Activity Diagram



Gambar 2. 7. Contoh Activity Diagram

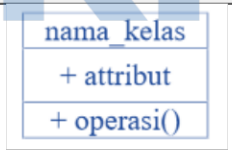
2.5.4 *Class Diagram*

Metodologi berorientasi objek bekerja untuk menemukan kelas, atribut, metode, dan hubungan antar kelas. Karena pemrograman terjadi di tingkat kelas, mendefinisikan kelas adalah salah satunya tugas analisis berorientasi objek paling penting. Diagram kelas menunjukkan fitur statis sistem dan tidak mewakili proses tertentu, diagram kelas juga menunjukkan sifat dari hubungan antar kelas [14].

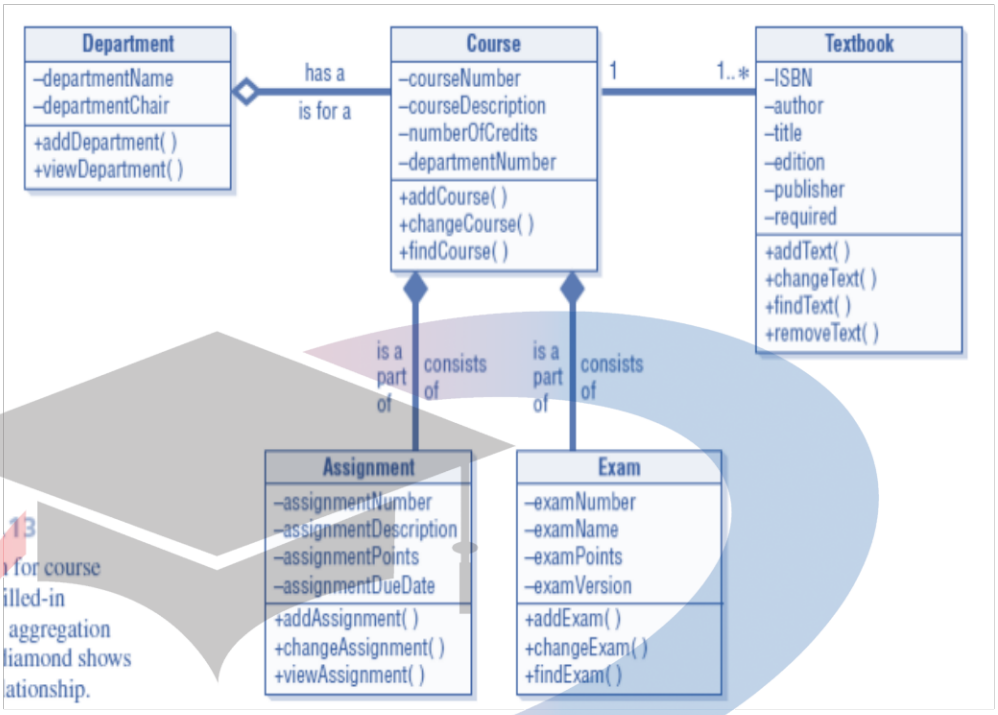
Kelas diwakili oleh persegi panjang pada diagram kelas. Dalam format yang paling sederhana, persegi panjang mungkin hanya menyertakan nama kelas, tetapi juga dapat menyertakan atribut dan metode. Atribut adalah apa yang diketahui kelas tentang karakteristik objek, atau metode. Metode adalah bagian kecil dari kode yang berfungsidengan atribut. mengilustrasikan diagram kelas untuk penawaran kursus. Perhatikan bahwa namanya berada di tengah di bagian atas kelas, biasanya dalam huruf tebal. Area tepat di bawah nama menunjukkan atribut, dan bagian bawah mencantumkan metode. Diagram kelas menunjukkan persyaratan penyimpanan data serta pemrosesan [14].

Class Diagram menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. *Class Diagram* mendeskripsikan jenis-jenis objek dalam sistem dan berbagai hubungan statis yang terdapat diantara mereka [13]. Berikut Simbol- simbol *Class Diagram* [13]:

Tabel 2. 2. Simbol-Simbol pada Class Diagram

No	Simbol	Keterangan	Definisi
1.		*Class*	Kelas pada struktur sistem
2.		*Interface*	Sama dengan konsep *interface* dalam pemrograman berorientasi objek

3.		*Association*	Relasi antar kelas dengan makna umum asosiasi biasanya juga disertai dengan *multiplicity*
4.		*Directed Association*	Relasi antar kelas dengan makna kelas yang atau digunakan oleh kelas yang lain, asosiasi biasanya juga disertai dengan *multiplicity.*
5.		*Generalisasi*	Relasi antar kelas dengan makna generalisasi-spesialisasi (umum khusus)
6.		*Dependency*	Relasi antar kelas dengan makna kebergantungan antarkelas
7.		*Aggregation*	Relasi antar kelas dengan makna semua-bagian (*whole-part*)



Gambar 2. 8. Ilustrasi Class Diagram

2.6 Basis Data

Basis data terdiri atas 2 kata, yaitu basis dan data. Basis kurang lebih dapat diartikan sebagai markas atau Gudang tempat berkumpul sedangkan data adalah representasi fakta dunia nyata yang mewakili suatu objek seperti manusia (pegawai, siswa, pembeli, pelanggan), barang, hewan, peristiwa, konsep, keadaan, dan sebagaiannya yang diwujudkan dalam bentuk angka, huruf, *symbol*, teks, gambar, bunyi, atau kombinasinya [15].

Basis data adalah sekumpulan data yang berelasi secara logikal dan deskripsi dari data dirancang untuk memenuhi kebutuhan informasi dari sebuah organisasi. Basis data merupakan tempat penyimpanan data tunggal yang dapat digunakan secara bersama-sama oleh banyak departemen dan pengguna. Dari pada menggunakan *file-file* yang berulang dan sama sekali tidak terhubung, basis data menyimpan semua data yang terintegrasi dengan jumlah duplikasi data seminimal mungkin [16].

Tujuan basis data yang efektif, yaitu [16]:

1. Memastikan bahwa data dapat dipakai di antara pemakai untuk berbagai aplikasi.

2. Memelihara data baik keakuratan maupun kekonsistenannya.
3. Memastikan bahwa semua data yang diperlukan untuk aplikasi sekarang dan yang akan datang akan disediakan dengan cepat.
4. Membolehkan basis data untuk berkembang dan kebutuhan pemakai untuk berkembang.
5. Membolehkan pemakai untuk membangun pandangan personalnya tentang data tanpa memperhatikan cara data disimpan secara fisik

Sistem basis data adalah suatu sistem untuk menyusun dan mengelola *record-record* dengan menggunakan komputer untuk menyimpan atau merekam serta memelihara data operasional lengkap sebuah organisasi atau perusahaan sehingga mampu menyediakan informasi yang optimal yang diperlukan pemakai untuk proses pengambilan keputusan [14].

Sebuah bahasa basis data biasanya dapat dipisah ke dalam dua bentuk, yaitu [16]:

1. *Data Definition Language (DDL)*

   Data Definition Language (DDL) adalah sebuah bahasa yang mengizinkan *Database Administrator (DBA)* atau pengguna untuk menjelaskan dan memberi nama pada entitas, atribut, dan hubungan yang diperlukan untuk aplikasi, secara bersamaan dengan *associated integrity* dan *security contraint*. *DDL* mengizinkan pengguna untuk menentukan tipe, struktur, serta kendala data yang nantinya akan disimpan ke dalam basis data.

2. *Data Manipulation Language (DML)*

   Data Manipulation Language (DML) adalah sebuah bahasa yang menyediakan satu set operasi untuk mendukung dasar dari operasi manipulasi pada data yang disimpan pada basis data. *DML* mengizinkan pengguna untuk menambah, mengubah, menghapus, dan mengambil data dari basis data tersebut. Bahasa standar dari *DBMS* ialah *Structured Query Language (SQL)*. Komponen utama dalam *DBMS* adalah sebagai berikut [14]:

   a. Data

b. Data di dalam sebuah basis data dapat disimpan secara terintegrasi dan data dapat dipakai secara bersama-sama.

c. Perangkat keras *(Hardware)*

Terdiri dari semua peralatan komputer yang digunakan untuk pengolahan system basis data, berupa peralatan untuk penyimpanan basis data (*secondary storage* seperti *disk*), peralatan *input* dan *output,* serta peralatan komunikasi data.

d. Perangkat lunak *(Software)*

Berfungsi sebagai perantara *(interface)* antara pemakai dengan data fisik pada basis data. *Software* pada basis data dapat berupa:

a) *Database Management System (DBMS)* yang menangani akses terhadap basis data sehingga pemakai tidak perlu memikirkan proses penyimpanan dan pengolahan data secara detail.

b) Program-program aplikasi dan prosedur-prosedur.

e. *User*

Database Administrator (DBA) merupakan orang atau tim yang bertugas mengelola sistem basis data secara keseluruhan. *DBA* mempunyai tugas mengontrol *DBMS* dan *software-software*, memonitor siapa yang mengakses basis data, mengatur pemakaian basis data, memeriksa keamanan, integrasi, *recovery*, atau cadangan data, serta persetujuan.

a) *Programmer,* merupakan orang atau tim yang bertugas membuat program aplikasi, misalnya untuk perbankan atau administrasi.

b) *End-user*, merupakan orang yang mengakses basis data melalui terminal dengan menggunakan bahasa *query* atau program aplikasi yang dibuat oleh *programmer*

2.7 Petshop

PetShop merupakan bisnis yang bergerak di bidang penjualan, pendistribusian dan pelayanan untuk hewan peliharaan. Pecinta binatang atau mereka yang peduli

terhadap binatang saat ini memberikan celah pada bidang bisnis *Pet Shop*. Kegiatan bisnis *PetShop* masih menggunakan proses konvensional berupa toko fisik pada saat konsumen ingin memesan produknya. Media pemasaran yang masih dilakukan secara konvensional tentunya kurang efektif dilakukan di era globalisasi ini. Teknologi dapat dimanfaatkan dan berperan penting untuk memenuhi tujuan bisnis ataupun digunakan untuk membagikan informasi yang penting [17].

Petshop sebagai salah satu bagian dari pelaku bisnis dalam bidang hewan peliharaan juga dituntut agar berkembang dalam penerapan teknologi. Dengan diterapkannya teknologi dalam proses bisnis tentu akan sangat membantu *petshop* dalam meningkatkan kinerjanya. Sistem akan mampu bekerja sesuai dengan kebutuhan yang dihadapi oleh *petshop* sehingga memberikan pelayanan dan kemudahan yang secara langsung berdampak pada peningkatan kualitas kinerja *petshop*. Secara umum *petshop* merupakan tempat untuk mencari kebutuhan dan keperluan[18].