

## **BAB II**

### **TINJAUAN PUSTAKA**

#### **II.1. Penelitian Terkait**

Adapun penelitian terkait yang akan digunakan sebagai sumber acuan yang relevan dan terkini yaitu:

Berdasarkan penelitian yang dilakukan oleh Mamay Syani (2016) dengan judul “Perancangan Aplikasi Pemesanan *Catering* Berbasis *Mobile Android*” Pemesanan *catering* merupakan suatu aktivitas yang dilakukan pelanggan untuk membeli produk berupa paket makanan. Di Cimahi *Catering* proses pemesanan *catering* dilakukan oleh pelanggan dengan datang langsung ke lokasi atau menelpon ke Cimahi *Catering*. Dalam hal pencatatan data pemesanan pun dirasa kurang efektif karena pendataannya yang masih bersifat konvensional. Hal ini beresiko terjadinya kesalahan pendataan. Sebagai solusi dari permasalahan tersebut penulis membangun sebuah Aplikasi Pemesanan *Catering* Berbasis *Mobile Android* untuk membantu dalam proses pemesanan *catering* oleh pelanggan serta pengelolaan data pemesanan *catering* oleh pihak Cimahi *Catering*.

Berdasarkan penelitian yang dilakukan Kusuma dan Prasetya (2017) dengan judul Perancangan Dan Implementasi *E-Commerce* Untuk Penjualan Baju Online Berbasis *Android*, menyimpulkan bahwa pengujian metode *whitebox* pada pembahasan diperoleh hasil kepuasan pelanggan sistem *e-commerce* penjualan baju sebesar 70%.

Berdasarkan penelitian yang dilakukan oleh Nugraha (2017) dengan judul Aplikasi Berbasis Android untuk Penjualan Paket Wisata Domestik Dan Internasioanl Menggunakan SMS Gateway, menyimpulkan bahwa Aplikasi dapat menampilkan pemesanan paket wisata yang berasal dari *website* (<http://www.traveliawisata.com>) menjadi bentuk yang sesuai pada perangkat *smartphone*.

Berdasarkan penelitian yang dilakukan Rendy Ryan, dkk (2018) dengan judul Aplikasi Pemesanan Air Mineral Berbasis Android Pada PT. Citra Golden Tunggal Pangkalpinang, Pada penelitian ini, dibangun sebuah aplikasi pada telepon seluler pintar berbasis Android. Tujuannya adalah untuk menjawab kebutuhan akan teknologi yang memudahkan dalam berbelanja secara praktis dan *mobile*. Model pengembangan aplikasi dengan menggunakan waterfall, dan menggunakan metode berorientasi objek untuk pengembangan aplikasinya. Pengujian dilakukan dengan menggunakan teknik *Blackbox*. Hasil yang diharapkan dengan adanya aplikasi ini dapat mempermudah pelanggan untuk melakukan pemesanan.

Berdasarkan penelitian dari Roberto kaban (2019) dengan judul “Aplikasi Pemesanan Tiket Bus Berbasis Android (Study Kasus: PT. Als Terminal Pasar X Tanjung Beringin)” Penelitian ini bertujuan untuk merancang aplikasi pemesanan tiket agar memudahkan petugas loket dan calon penumpang dalam proses pemesanan tiket. Dalam tahap pengumpulan data, penulis melakukan studi lapangan, observasi langsung dan wawancara pada petugas loket PT. ALS Terminal Pasar X Tanjung Beringin. Perancangan aplikasi ini menggunakan

metode desain berorientasi objek dengan UML (Unified Modeling Language). Aplikasi berbasis android ini dibangun dengan bahasa pemrograman Java menggunakan Android studio dan pada petugas loket untuk mengelola pemesanan tiket menggunakan bahasa pemrograman PHP, Bootstrap sebagai interface dan MariaDB sebagai database.

Berdasarkan penelitian dari Janifer W Janis, dkk (2020) dengan judul “Rancang Bangun Aplikasi Online Sistem Pemesanan Jasa Tukang Bangunan Berbasis Lokasi” Tujuan penelitian ini yaitu merancangan aplikasi online sistem pemesanan jasa tukang bangunan berbasis lokasi untuk mendukung permasalahan kebutuhan masyarakat dan tukang bangunan dalam pekerjaan pembangunan. Dengan adanya aplikasi ini, masyarakat (selaku pemesan jasa) dapat dengan mudah melakukan pemesanan jasa tukang bangunan berdasarkan lokasi secara online (terkoneksi internet).

Adapun perbedaan penelitian yang dilakukan penulis dengan penelitian terkait sebelumnya yaitu pada penelitian ini penulis menggunakan kombinasi web dan *Android* untuk melakukan pemesanan obat pada Apotik Mitha Farma. Sehingga dengan adanya aplikasi ini, dapat memudahkan *customer* dalam melakukan pemesanan obat dan mempercepat proses penjualan pada Apotik Mitha Farma.

## II.2. Landasan Teori

### II.2.1. Aplikasi

Pengertian aplikasi adalah suatu bagian dari perangkat lunak yang dapat digunakan untuk menyelesaikan masalah-masalah yang khusus yang dihadapi *user* dengan menggunakan kemampuan komputer. (Fergiawan Listianto, dkk : 2017)

Aplikasi adalah program siap pakai yang dapat digunakan untuk menjalankan perintah-perintah dari pengguna aplikasi tersebut dengan tujuan mendapatkan hasil yang lebih akurat sesuai dengan tujuan pembuatan aplikasi tersebut, aplikasi mempunyai arti yaitu pemecahan masalah yang menggunakan salah satu teknik pemrosesan data aplikasi yang biasanya berpacu pada sebuah komputasi yang diinginkan atau diharapkan maupun pemrosesan data yang diharapkan. (Ari Setiaji, dkk : 2020).

### II.2.2. Android

*Android* adalah suatu sistem operasi yang berbasis *Linux* dengan perangkat *Mobile device*, *Android* menyediakan platform terbuka bagi para pengembang untuk menciptakan aplikasi-aplikasi baru untuk membantu kebutuhan manusia. Awal terdirinya android dimulai dari *Google Inc* yang membeli *Android inc* pendatang baru dari perangkat lunak untuk ponsel/*smartphone*. Semakin banyak *smartphone* dan *PC Tablet* menggunakan sistem operasi dengan versi yang berbeda. Semakin tinggi versinya maka fiturnya semakin canggih dan banyak. Telepon pertama yang memakai sistem operasi *Android* adalah *HTC Dream* yang

dirilis pada tanggal 22 Oktober 2008. Perkembangan sistem *Android* berkembang sesuai versi keluarnya adapun versi *Android* dimulai dari (Muhamad Zaelani ; 2017: 5)

**Tabel. II.1 Versi Android**

| Versi   | Nama                            | Rilis                                                             | Catatan                                                                                                                         |
|---------|---------------------------------|-------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------|
| 1.0     | <i>Android 1.0</i>              | 23 September 2008                                                 | <i>Android</i> pertama hanya untuk <i>smartphone</i>                                                                            |
| 1.1     | <i>Android 1.1</i>              | 9 Februari 2008                                                   |                                                                                                                                 |
| 1.5     | <i>Cupcake</i>                  | 30 April 2009                                                     | Mulai pakai kode nama                                                                                                           |
| 1.6     | <i>Donut</i>                    | 15 September 2009                                                 |                                                                                                                                 |
| 2.0-2.1 | <i>Éclair</i>                   | 26 Oktober 2009 (2.0)<br>12 Januari 2010 (2.1)                    |                                                                                                                                 |
| 2.2     | <i>Froyo</i>                    | 20 Mei 2010                                                       |                                                                                                                                 |
| 2.3     | <i>Gingerbread</i>              | 6 Desember 2010                                                   | Digunakan pada <i>smartphone</i> jenis lama                                                                                     |
| 3.0-3.2 | <i>Honeycomb</i>                | 22 Februari 2011 (3.0)<br>10 Mei 2011 (3.1)<br>15 Juli 2011 (3.2) | Hanya untuk tablet                                                                                                              |
| 4.0     | <i>ICS (Ice Cream Sandwich)</i> | 19 Oktober 2011                                                   | <i>Smartphone</i> dan tablet                                                                                                    |
| 4.1-4.3 | <i>Jelly Bean</i>               | 9 Juli 2012 (4.1)<br>13 November 2012 (4.2)<br>24 Juli 2013 (4.3) | <i>Update</i> untuk memperbaiki dan menambah <i>fitur</i> pada <i>ICS</i>                                                       |
| 4.4     | <i>Kit kat</i>                  | 3 September 2013                                                  |                                                                                                                                 |
| 5.0     | <i>Lollipop</i>                 | 12 november 2014 (5.0)<br>9 Maret 2015 (5.1)                      |                                                                                                                                 |
| 6.0     | <i>Marshmallow</i>              | 5 oktober 015                                                     | Terdapat <i>daze mode</i> , <i>Do Not Disturb mode</i> , mendukung <i>USB tipe C</i> , mendukung pembacaan <i>fingerprint</i> . |
| 7.0     | <i>Nougat</i>                   | 22 Agustus 2016                                                   | Multitasking                                                                                                                    |

Sumber: Muhamad Zaelani (2017: 5)

Adapun penjelasan dari versi as android diatas adalaha sebagai berikut :

#### 1. *Android 1.0: Astro (Alpha)*

Kedua versi awal *Android* ini mungkin agak asing kamu dengarkan. Pasalnya versi *Android 1.0 Astro (Alpha)* dan *Android 1.1* Walaupun belum menggunakan nama makanan manis, kedua sistem operasi *Android* ini tentu menjadi *pionir*. Pasalnya di sinilah *Android* bermula lewat *smartphone* pertama, *HTC Dream*.

#### 2. *Android 1.1 : Bender (Beta)*

*Android 1.1 Bender (Beta)* ini belum diluncurkan secara publik untuk kebutuhan komersil. *Platform Android* sendiri pertama kali diluncurkan pada September 2008 dengan andil Andy Rubin yang saat ini dikenal sebagai Bapak *Android*. Walaupun belum menggunakan nama makanan manis, kedua sistem operasi *Android* ini tentu menjadi *pionir*. Pasalnya di sinilah *Android* bermula lewat *smartphone* pertama, *HTC Dream*. (Muhamad Zaelani ; 2017: 5)

#### 3. *Cupcake*

Mulai dari versi ini, *Android* menggunakan nama makanan manis untuk setiap versi yang diluncurkan. *Android 1.5 Cupcake* sendiri dirilis pada tanggal 30 April 2009 dengan berbagai fitur di sebuah perangkat *smartphone* untuk menggantikan *featured phone* kala itu. (Muhamad Zaelani ; 2017: 5)

#### 4. *Donut*

Tentu pada awal perilisannya, sistem operasi *Android* tetap memiliki banyak *bug* yang pengembangnya perlu mengadakan perbaikan. Hal ini dilakukan pada *Android 1.6 Donut* yang dirilis pada 15 September 2009. Ya artinya belum

genap setahun semenjak perilisan *Android 1.5 Cupcake* atau hanya berselang 5 bulan saja. *Android* pun menambahkan beberapa pembaruan, terutama dukungan pada layar *smartphone* yang lebih besar. (Muhamad Zaelani ; 2017: 5)

#### 5. *Eclair*

Sama seperti versi sebelumnya, *Android 2.0 & 2.1 Eclair* masih berfungsi untuk menutupi *bug* yang masih ditemukan pada sistem operasi *mobile* ini. Di samping itu, *Android* juga menambah berbagai fitur di dalamnya.

Mulai dari dukungan *Bluetooth* hingga fitur kamera yang mulai menjadi nilai jual *smartphone* kala itu. *Android 2.0 & 2.1 Eclair* digunakan pada perangkat seperti *HTC Nexus One*. (Muhamad Zaelani ; 2017: 5)

#### 6. *Android 2.2 : Froyo (Frozen Yoghurt)*

Mulai versi ini *Android* tampaknya sudah mulai dikenal luas oleh berbagai *brand smartphone*. *Android 2.2 Frozen Yoghurt* alias *Froyo* ini dirilis pertama kali pada tanggal 20 Mei 2010. Walaupun sudah mulai dipergunakan pada beberapa *brand*, namun tetap saja *Android* masih kalah bersaing dengan *Symbian* yang mendominasi pasar *featured phone*. *Android 2.2 Froyo* memberikan peningkatan pada kecepatan kerja, fitur *USB tethering*, *WiFi hotspot*, serta fitur keamanan. (Muhamad Zaelani ; 2017: 5)

#### 7. *Gingerbread*

Belum setahun berselang lagi, *Android 2.3 Gingerbread* kembali diluncurkan pada Desember 2010 dengan berbagai peningkatan yang cukup signifikan. Hal ini terutama pada tampilan tatap muka alias *user interface* yang digunakan. Mulai versi ini, banyak *brand smartphone* mulai melirik

menggunakan sistem operasi *Android*. Salah satunya *Samsung Galaxy Series* yang populer hingga saat ini. (Muhamad Zaelani ; 2017: 5)

#### 8. *Honeycomb*

Untuk para pengguna *smartphone* mungkin akan agak asing dengan versi *Android* yang satu ini. *Android 3.0 & 3.2 Honeycomb* yang menggunakan ikon lebah ini memang diperuntukkan penggunaannya khusus untuk perangkat *tablet*. Tentu perilisan *Android 3.0 & 3.2 Honeycomb* pada 10 Mei 2011 ini untuk mendukung *Samsung* yang mulai merilis perangkat *tablet*, *Samsung Galaxy Tab Series* untuk menyaingi *Apple iPad*. (Muhamad Zaelani ; 2017: 5)

#### 9. *Ice Cream Sandwich (ICS)*

Selain itu, *Android* pun juga merilis versi *Android 4.0 Ice Cream Sandwich* yang kembali diperuntukkan untuk perangkat *smartphone*. *Android 4.0 Ice Cream Sandwich* sendiri dirilis pada 19 Oktober 2011 silam. Punya versi nama paling panjang hingga saat ini, *Android 4.0 Ice Cream Sandwich* memberikan banyak pembaruan. Mulai dari animasi yang semakin halus, sederhana, dan mudah digunakan. (Muhamad Zaelani ; 2017: 5)

#### 10. *Jelly Bean*

Peningkatan signifikan terasa saat menggunakan *Android 4.1 & 4.3 Jelly Bean*. Sistem operasi ini sendiri pertama kali dirilis pada Juni 2012 dengan membawa sejumlah peningkatan terutama di sektor pengolahan grafis. Dengan begini, tentu *Android 4.1 & 4.3 Jelly Bean* bisa memberikan peningkatan fungsi pada *user interface* dan teknologi *Vsync* yang digunakannya. (Muhamad Zaelani ; 2017: 5)

### 11. *KitKat*

Menggunakan nama *brand* cemilan terkenal, *Android 4.4 KitKat* pertama kali dirilis pada Oktober 2013. Versi *Android* terbaik ini pun bisa dikatakan menjadi favorit karena mendukung hampir seluruh perangkat *smartphone* di dunia. Hal ini dikarenakan *Android 4.4 KitKat* dapat memberikan optimalisasi yang baik termasuk pada perangkat *smartphone* yang memiliki *spesifikasi* kurang mumpuni alias cukup rendah saat itu. (Muhamad Zaelani ; 2017: 5)

### 12. *Lollipop*

Mulai beberapa versi ke belakang, *Android* dan *Google* pun mulai secara rutin memperbarui sistem operasi mereka dalam selang waktu setahun. Termasuk *Android 5.0 & 5.1 Lollipop* yang dirilis dan diresmikan pada Juni 2014. Bisa dibilang *Android 5.0 & 5.1 Lollipop* menjadi *pionir* dibuatnya *smartphone flagship* dengan *spesifikasi* cukup mumpuni. Versi *Android* ini sudah mendukung arsitektur 64-bit yang sudah memungkinkan penggunaan RAM di atas 3GB. Salah satunya *ASUS Zenfone 2* yang sudah mengusung RAM 4GB saat itu. (Muhamad Zaelani ; 2017: 5)

### 13. *Marshmallow*

*Android 6.0 Marshmallow* menjadi suksesor dari versi *Android* sebelumnya. Sistem operasi ini sendiri pertama kali diperkenalkan pada Mei 2015 dan mulai dirilis pada Oktober 2015 silam. Sistem operasi ini secara jelas memberikan peningkatan pada sistem keamanan dengan dihadirkannya *fingerprint sensor* sebagai sistem keamanan *biometrik* yang digunakan. Selain digunakan untuk mengunci layar, *fingerprint sensor* ini dapat digunakan untuk otentikasi

*Google Play Store* dan pembelian dengan menggunakan *Android Pay*. (Muhamad Zaelani ; 2017: 5)

#### 14. *Nougat*

Untuk saat ini, sistem operasi *Android* ini masih digunakan beberapa *smartphone* yang baru dirilis belakangan ini. *Android 7.0 & 7.1 Nougat* pertama kali diperkenalkan pada Juni 2016 dengan menampilkan ikon robot *Android* dengan batangan *Nougat*. Sistem operasi *Android 7.0 & 7.1 Nougat* mengalami perubahan dari segi tampilan antarmuka. Selain itu ada juga fitur *splitscreen* untuk membagi tampilan layar untuk dua aplikasi sekaligus. (Muhamad Zaelani ; 2017: 5)

### II.2.3. *Android Studio*

*Android Studio* adalah Lingkungan Pengembangan Terpadu - Integrated Development Environment (IDE) untuk pengembangan aplikasi *Android*, berdasarkan IntelliJ IDEA. Konsep yang dimiliki *Android Inc* ternyata menggugah minat *Google* untuk memilikinya. Pada bulan Agustus 2005, Akhirnya *Android Inc* diakuisisi oleh *Google Inc.* seluruh sahamnya dibeli oleh *Google*. Banyak yang memperkirakan nilai pembelian *Android Inc* Oleh *Google* adalah sebesar USD 50 juta. saat itu banyak yang berspekulasi bahwa akuisisi ini adalah langkah awal yang dilakukan *Google* untuk masuk kepasar mobile phone. Maka dari situlah banyak pengembang sistem maupun software berlomba lomba untuk membuat atau merancang sistem *Android* menggunakan software – software yang support dengan *Android*, dan sebagai contoh disini kita pengenalan

*Android Studio*. Selain merupakan editor kode IntelliJ dan alat pengembang yang berdaya guna, *Android Studio* menawarkan fitur lebih banyak untuk meningkatkan produktivitas Anda saat membuat aplikasi *Android*, misalnya:

1. Sistem versi berbasis *Gradle* yang *fleksibel*
2. Emulator yang cepat dan kaya fitur
3. Lingkungan yang menyatu untuk pengembangan bagi semua perangkat *Android*
4. Instant Run untuk mendorong perubahan ke aplikasi yang berjalan tanpa membuat APK baru
5. Template kode dan integrasi GitHub untuk membuat fitur aplikasi yang sama dan mengimpor kode contoh
6. Alat pengujian dan kerangka kerja yang ekstensif
7. Alat Lint untuk meningkatkan kinerja, kegunaan, kompatibilitas versi, dan masalah - masalah lain
8. Dukungan C++ dan NDK
9. Dukungan bawaan untuk *Google Cloud Platform*, mempermudah pengintegrasian *Google Cloud Messaging* dan App Engine

Laman ini berisi pengantar dasar fitur-fitur *Android Studio*. Untuk memperoleh rangkuman perubahan terbaru, lihat Catatan Rilis *Android Studio* (*Google developer* ; 2017).

#### II.2.4 Basis Data (*Database*)

Basis data atau *database* merupakan koleksi dari data-data yang terorganisir dengan rapi sehingga data dapat dengan mudah disimpan dan dimanipulasi. Kita dapat menjumpai pemanfaatan *database* dalam kehidupan sehari-hari, seperti penggunaan mesin ATM, sistem akademi universitas/sekolah, sistem informasi penjualan. Salah satu tujuan dari *database* adalah memberikan pengguna suatu pandangan abstrak dari data, yaitu sistem menyembunyikan rincian bagaimana data disimpan dan dipelihara. Sistem *database* harus dibuat semudah mungkin untuk dimengerti karena kebanyakan pengguna sistem *database* adalah orang-orang yang kurang terlatih di bidang teknologi. (Ni Kadek Ceryna, dkk : 2018 : 102)

#### II.2.5. MySQL

*MySQL* adalah salah satu jenis *database server* yang sangat terkenal. Kepopulerannya disebabkan *MySQL* menggunakan *SQL* sebagai bahasa dasar untuk mengakses databasenya. *MySQL* termasuk jenis *RDBMS (Relational Database Management System)*. Pada *MySQL*, sebuah database mengandung satu atau sejumlah tabel. Tabel terdiri atas sejumlah baris dan setiap baris mengandung satu atau beberapa kolom. (Priyo Sutopo, dkk : 2017)

*MySQL (My Structure Query Language)* merupakan sebuah program pembuat database yang bersifat *Open Source*, artinya semua orang dapat menggunakannya dan dapat dijalankan pada semua *platform* baik *Windows* maupun *linux*. *MySQL* juga merupakan sebuah perangkat lunak sistem manajemen basis

data *SQL* yang bersifat jaringan sehingga dapat digunakan untuk aplikasi multi *user*. *MySQL* juga sering dikenal dengan nama sistem manajemen *database* relasional. Suatu *database* relasional menyimpan data dalam table yang terpisah. Tabel -table tersebut terhubung oleh suatu relasi terdefinisi yang memungkinkan memperoleh kombinasi data dari beberapa table dalam suatu permintaan. Untuk administrasi *database*, seperti pembuatan *database*, pembuatan tabel, dan sebagainya dapat digunakan aplikasi berbasis web seperti *PHP MyAdmin* dengan aplikasi *XAMPP*. (Saipul Anwar, 2016)

### II.2.7. PHP

PHP atau kependekan dari *Hypertext Preprocessor* adalah salah satu bahasa pemrograman *open source* yang sangat cocok atau dikhususkan untuk pengembangan *web* dan dapat ditanamkan pada sebuah skripsi HTML. Bahasa PHP dapat dikatakan menggambarkan beberapa bahasa pemrograman seperti C, *Java*, dan *Perl* serta mudah untuk dipelajari. PHP merupakan bahasa *scripting server – side*, dimana pemrosesan datanya dilakukan pada sisi *server*. Sederhananya, *server* yang akan menterjemahkan skrip program, baru kemudian hasilnya akan dikirim kepada *client* yang melakukan permintaan. Adapun pengertian lain PHP adalah *akronim* dari *Hypertext Preprocessor*, yaitu suatu bahasa pemrograman berbasis kode – kode (*script*) yang digunakan untuk mengolah suatu data dan mengirimkannya kembali ke *web browser* menjadi kode HTML”. (Astria Firman, 2016 : 30).

PHP adalah salah satu server side yang dirancang khusus untuk aplikasi web. PHP disisipkan diantara bahasa HTML dan karena bahasa server side, maka bahasa PHP akan dieksekusi di server, sehingga yang dikirimkan ke browser adalah hasil jadi dalam bentuk HTML, dan kode PHP tidak akan terlihat. PHP termasuk Open Source Product. Jadi, dapat diubah source code dan mendistribusikanya secara bebas (Priyo Sutopo, dkk : 2017)

PHP (singkatan rekursif: *PHP Hypertext Preprocessor*) merupakan script yang dapat disisipkan ke dalam HTML. PHP dipakai untuk membuat sistem berbasis web yang dinamis. PHP bahasa pemrograman yang *server-side* sehingga program tersebut akan dijalankan atau diproses oleh server . Hingga saat ini PHP sudah rilis versi 7.4.1 pada 18 desember 2019. (Roberto kaban,. Dkk : 2019)

#### **II.2.8. UML (*Unified Modelling Language*)**

*Unified Modelling Language* (UML) merupakan satu kumpulan konvensi pemodelan yang digunakan untuk menentukan atau menggambarkan sebuah sistem software yang terkait dengan objek. UML merupakan salah satu alat bantu yang sangat handal dalam bidang pengembangan sistem berorientasi objek karena UML menyediakan Bahasa pemodelan visual yang memungkinkan pengembang sistem membuat *blue print* atas visinya dalam bentuk yang baku. UML berfungsi sebagai jembatan dalam mengkomunikasikan beberapa aspek dalam sistem melalui jumlah elemen grafis yang bisa dikombinasikan menjadi.

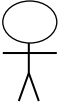
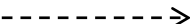
*Unified Modeling Language (UML)* biasa digunakan untuk :

- a. Menggambarkan batasan sistem dan fungsi - fungsi sistem secara umum, dibuat dengan *use case* dan *actor*.
- b. Menggambarkan kegiatan atau proses bisnis yang dilaksanakan secara umum, dibuat dengan *interaction diagrams*.
- c. Menggambarkan representasi struktur *static* sebuah sistem dalam bentuk *class diagrams*.
- d. Membuat model behavior “yang menggambarkan kebiasaan atau sifat sebuah sistem” dengan *state transition diagrams*.
- e. Menyatakan arsitektur implementasi fisik menggunakan *component and development*.
- f. Menyampaikan atau memperluas *functionality* dengan *stereotypes*. (Omni Alfina dan Fitriana Harahap : 2019)

#### **II.2.8.1. 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. Simbol-simbol yang digunakan dalam *use case* diagram dapat dilihat pada tabel II.2 di bawah ini :

Tabel II.2. Simbol *Use Case*

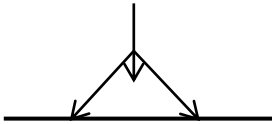
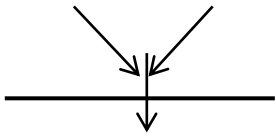
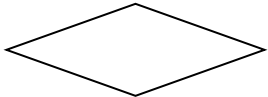
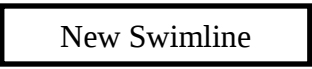
| Gambar                                                                              | Keterangan      | Deskripsi                                                                                                                                                                                                   |
|-------------------------------------------------------------------------------------|-----------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|    | <i>Use case</i> | Menggambarkan fungsionalitas yang disediakan sistem sebagai unit-unit yang bertukar pesan antar unit dengan aktor, biasanya dinyatakan dengan menggunakan kata kerja di awal nama <i>use case</i> .         |
|    | Aktor           | Sistem yang lain yang mengaktifkan fungsi dari target sistem. Untuk mengidentifikasi aktor, harus ditentukan pembagian tenaga kerja dan tugas-tugas yang berkaitan dengan peran pada konteks target sistem. |
|   | Asosiasi        | Penghubung antara aktor dan <i>use case</i> , digambarkan dengan garis tanpa panah yang mengindikasikan siapa atau apa yang meminta interaksi secara langsung dan bukannya mengindikasikan aliran data.     |
|  | Asosiasi        | Penghubung antara aktor dan <i>use case</i> yang menggunakan panah terbuka untuk mengindikasikan bila aktor berinteraksi secara pasif dengan sistem.                                                        |
|  | <i>Include</i>  | Merupakan di dalam <i>use case</i> lain ( <i>required</i> ) atau pemanggilan <i>use case</i> oleh <i>use case</i> lain, contohnya adalah pemanggilan sebuah fungsi program.                                 |
|  | <i>Extend</i>   | Merupakan perluasan dari <i>use case</i> lain jika kondisi atau syarat terpenuhi.                                                                                                                           |

(Sumber : Ade Hendini, 2016)

### II.2.8.2. Activity Diagram

*Activity diagram* menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis. Simbol-simbol yang digunakan dalam *activity diagram* dapat di lihat pada tabel II.3 dibawah ini:

Tabel II.3. Simbol *Activity Diagram*

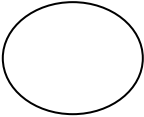
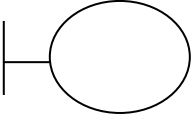
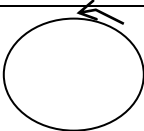
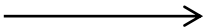
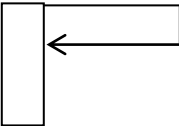
| Gambar                                                                              | Keterangan                    | Deskripsi                                                                                                                       |
|-------------------------------------------------------------------------------------|-------------------------------|---------------------------------------------------------------------------------------------------------------------------------|
|    | <i>Start point</i>            | Diletakkan pada pojok kiri atas dan merupakan awal aktifitas.                                                                   |
|    | <i>End point</i>              | Akhir aktifitas.                                                                                                                |
|    | <i>Activites</i>              | Menggambarkan suatu proses/kegiatan bisnis.                                                                                     |
|    | <i>Fork</i><br>(Percabangan). | Digunakan untuk menunjukkan kegiatan yang dilakukan secara parallel atau untuk menggabungkan dua kegiatan pararel menjadi satu. |
|  | <i>Join</i><br>(penggabungan) | Digunakan untuk menunjukkan adanya dekomposisi.                                                                                 |
|  | <i>Decision Points</i>        | Menggambarkan pilihan untuk pengambilan keputusan, <i>true</i> , <i>false</i> .                                                 |
|  | <i>Swimlane</i>               | Pembagian <i>activity</i> diagram untuk menunjukkan siapa melakukan apa.                                                        |

(Sumber : Ade Hendini, 2016 : 109)

### II.2.8.3. *Sequence Diagram*

*Sequence* diagram menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan pesan yang dikirimkan dan diterima antar objek. Simbol-simbol yang digunakan dalam *sequence diagram* dapat dilihat pada tabel II.4 di bawah ini :

Tabel II.4. Simbol *Sequence Diagram*

| Gambar                                                                              | Keterangan            | Deskripsi                                                                                                                                                               |
|-------------------------------------------------------------------------------------|-----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|    | <i>Entity Class</i>   | Merupakan bagian dari sistem yang berisi kumpulan kelas berupa entitas-entitas yang membentuk gambaran awal sistem dan menjadi landasan untuk menyusun basis data.      |
|    | <i>Boundary Class</i> | Berisi kumpulan kelas yang menjadi <i>interface</i> atau interaksi antara satu atau lebih aktor dengan sistem, seperti tampilan formentry dan <i>form</i> cetak.        |
|   | <i>Control class</i>  | Suatu objek yang berisi logika aplikasi yang tidak memiliki tanggung jawab kepada entitas, contohnya adalah kalkulasi dan aturan bisnis yang melibatkan berbagai objek. |
|  | <i>Message</i>        | Simbol mengirim pesan antar <i>class</i> .                                                                                                                              |
|  | <i>Recursive</i>      | Menggambarkan pengiriman pesan yang dikirim untuk dirinya sendiri.                                                                                                      |
|  | <i>Activation</i>     | Mewakili sebuah eksekusi operasi dari objek, panjang kotak ini berbanding lurus dengan durasi aktivitas sebuah operasi.                                                 |
|  | <i>Lifeline</i>       | Garis titik-titik yang terhubung dengan objek, sepanjang <i>lifeline</i> terdapat <i>activation</i> .                                                                   |

(Sumber : Ade Hendini, 2016 : 110)

#### II.2.8.4. 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. *Class diagram* juga menunjukkan atribut-atribut dan operasi-operasi dari sebuah kelas dan *constraint* yang berhubungan dengan objek yang dikoneksikan. *Class diagram* secara khas meliputi : Kelas (*Class*), Relasi *Associations*, *Generalization* dan *Aggregation*, atribut (*Attributes*), operasi (*operation/method*) dan *visibility*, tingkat akses objek eksternal kepada suatu operasi atau atribut. Hubungan antar kelas mempunyai keterangan yang disebut dengan *Multiplicity* atau *Cardinality* (Ade Hendini, 2016 : 111).

**Tabel II.5. Multiplicity Class Diagram**

| <b>Multiplicity</b> | <b>Penjelasan</b>                                               |
|---------------------|-----------------------------------------------------------------|
| 1                   | Satu dan hanya satu                                             |
| 0..*                | Boleh tidak ada atau 1 atau lebih                               |
| 1..*                | 1 atau lebih                                                    |
| 0..1                | Boleh tidak ada, maksimal 1                                     |
| n..n                | Batasan antara. Contoh 2..4 mempunyai arti minimal 2 maksimum 4 |

(Sumber : Ade Hendini, 2016 : 110)