# BAB II

# TINJAUAN PUSTAKA

## II.1. Penelitian Terdahulu

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

Berdasarkan penelitian dari Sidik Priyo Utomo (2020) dengan judul "Perancangan *RESTFul Web Service* Pada Sistem Informasi Terintegrasi Menggunakan *FrameWork CodeIgniter*" sistem Informasi Perguruan Tinggi Terintegrasi merupakan gabungan beberapa sistem yang berada dalam sebuah lingkungan Perguruan Tinggi. Sistem Informasi Perguruan Tinggi Terintegrasi dikembangkan menggunakan *REST web service*. Penelitian ini melakukan uji coba proses integrasi 2 server menggunakan RESTFul Web Service menggunakan *framework CodeIgniter* dan dapat diketahui hasil uji response time menggunakan aplikasi Postman. Dari hasil pengujian didapatkan sebuah grafik pengujian GET data dengan hasil Jumlah data dengan responsetime yang berbanding lurus

Berdasarkan penelitian dari Randi Rizal (2019) Penerapan teknologi web service dengan menggunakan arsitektur REST pada sistem informasi akademik dan sistem informasi perpustakaan mampu mengintegrasikan kedua sistem tersebut. Sehingga proses input dan verifikasi data hanya dilakukan satu kali, hal tersebut mengatasi terjadinya duplikasi data dan mengurangi perkerjaan input data. Format pertukaran data antar sistem menggunakan format JSON yang tidak bernegara (stateless), sehingga

memudahkan untuk dapat diakses oleh bahasa pemrograman apapun, arsitektur maupun sistem operasi yang berbeda.

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 Aplkasi 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).

## II.2. Landasan Teori

Untuk mendukung keberhasilan penelitian ini, penyusun melakukan pendekatan teoritis melalui beberapa literatur yang berhubungan dengan penelitian yang dilakukan. Beberapa tinjauan pustaka pada penelitian ini yaitu:

### II.2.1. *Restful Web Service*

*Restful Web Service* yang merupakan gaya arsitektur untuk penerapan *web service* dalam menerapkan konsep perpindahan antar state. *Restful Web Service* popular karena kesederhanaannya, sebagai lawan menciptakan standar baru, kerangka kerja dan teknologi. Keuntungan *Restful Web Service* : interaksi berbasis REST menggunakan Hypertext Transfer Protocol (HTTP) Internet yang merupakan hal yang sudah umum digunakan. Contoh dari pengaturan ini adalah interaksi berbasis REST semua berkomunikasi status mereka

menggunakan kode status HTTP standar. Jadi, 404 berarti sumber daya yang diminta tidak ditemukan; kode 401 berarti permintaan tidak diotorisasi; kode 200 berarti semuanya OK; dan 500 berarti ada kesalahan aplikasi yang tidak dapat dipulihkan pada server. (Randi Rizal : 2019)

### II.2.1. *Android*

Sistem Aplikasi *Android* merupakan sebuah sistem operasi yang berbasis Linux untuk telepon Seluler seperti telepon pintar dan komputer tablet. Sistem operasi ini bersifat *open source* (terbuka) sehingga para *programmer* dapat membuat aplikasi secara mudah untuk digunakan oleh bermacam perangkat bergerak (misalnya telepon seluler). Karena sistem operasi *Android* ini merupakan aplikasi *open source* maka dapat dilakukan modifikasi dan penyebaran secara bebas. (Yusfrizal Bin : 2018)

### II.2.2 *Android* Studio

*Android* Studio merupakan subset perangkat lunak untuk perangkat *mobile* yang meliputi system operasi, *middleware* dan aplikasi inti yang direlease oleh *Google. Android SDK* (*Software Development Kit*) menyediakan *Tools* dan *API* yang diperlukan untuk mengembangkan aplikasi pada *platform Android* dengan menggunakan bahasa pemrograman Java. Sejarah *Android* diawali tahun 2005 *Google* kemudian pada tahun itu juga memulai membangun *platform Android* secara intensif. 12 November 2007 *Google* bersama *OHA* (*Open Handset*

*Alliance*) yaitu konsorsium perangkat *mobile* terbuka, merilis *Google Android* SDK, setelah mengumumkan seminggu sebelumnya. (Faris Sifauttijani: 2017)

### II.2.2.1. Sejarah *Android*

*Android* adalah sistem operasi yang berbasis *Linux* untuk telepon seluler seperti telepon pintar dan komputer tablet. Android menyediakan *platform* terbuka bagi para pengembang untuk menciptakan aplikasi mereka sendiri untuk digunakan oleh bermacam peranti bergerak. Awalnya, *Google Inc.* membeli *Android Inc*., pendatang baru yang membuat peranti lunak untuk ponsel. Kemudian untuk mengembangkan Android, dibentuklah *Open Handset Alliance*, konsorsium dari 34 perusahaan peranti keras, peranti lunak, dan telekomunikasi, termasuk *Google, HTC, Intel, Motorola, Qualcomm, T-Mobile,* dan *Nvidia.*

Pada saat perilisan perdana Android, 5 November 2007, Android bersama *Open Handset Alliance* menyatakan mendukung pengembangan standar terbuka pada perangkat seluler. Di lain pihak, *Google* merilis kode–kode Android di bawah *lisensi Apache*, sebuah *lisensi* perangkat lunak dan standar terbuka perangkat seluler. (Harni Kusniyati, dkk : 2016).

*Android* SDK merupakan tools bagi para *programmer* yang ingin mengembangkan aplikasi berbasis *google Android*. *Android* SDK mencakup seperangkat alat pengembangan yang komprehensif. *Android* SDK terdiri dari *debugger, libraries, handset emulator*, dokumentasi, contoh kode, dan tutorial.

### II.2.2.2. Urutan Versi OS *Android*

**1. *Android* 1.0 & 1.1 : *Astro (Alpha)* & *Bender (Beta)***

Kedua versi awal *Android* ini mungkin agak asing kamu dengarkan. Pasalnya versi *Android* 1.0 *Astro (Alpha)* dan *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*.

**2. *Android* 1.5 : *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.

**3. *Android* 1.6 : *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.

**4. *Android* 2.0 & 2.1 *: 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.*

**5. *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.

**6. *Android* 2.3: *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.

**7. *Android* 3.0 & 3.2*: 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*.

**8. *Android 4.0 : Ice Cream Sandwich***

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.

**9. *Android* 4.1 & 4.3 : *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.

**10. *Android* 4.4 : *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.

**11. *Android* 5.0 & 5.1: *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 memungkinan penggunaan RAM di atas 3GB. Salah satunya *ASUS Zenfone* 2 yang sudah mengusung RAM 4GB saat itu.

**12. *Android* 6.0 : *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*.

**13. *Android* 7.0 & 7.1 : *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.

### *14. Android* 8.0 & 8.1 : *Oreo*

*Android* 8.0 & 8.1 *Oreo* menjadi sistem operasi *Android* paling terbaru yang banyak digunakan saat ini. Sistem operasi ini dirilis secara stabil mulai Agustus 2017 sudah mengalami pembaruan lewat versi *Android* 8.1 *Oreo*. *Android* 8.0 & 8.1 *Oreo* menjadi versi *Android* kedua yang menggunakan makanan manis dari nama *brand* terkenal setelah *Android* 4.4 *KitKat*. Sistem operasi ini menawarkan pengalaman *multitasking* yang makin mumpuni dibanding versi sebelumnya.

Selain itu ada juga *Project Treble* yang memungkinkan pengguna mendapat pembaruan lebih cepat.

### 15. *Android* 9.0 : *Pie*

Kemudian ada *Android* 9.0 *Pie* yang secara resmi diperkenalkan pada Agustus 2018. Sistem operasi *Android* ini memberi banyak perubahan, terutama untuk HP dengan desain baru. Misal *Android* 9.0 *Pie* memberikan navigasi berupa *gesture* yang menggantikan tombol fisik *Home*, *Back*, dan *Recent Apps*. Fitur lainnya yang cukup berguna adalah sistem notifikasi, pengatur kecerahan, hingga sistem *screenshoot* terbaru yang lebih memudahkan.

**16. *Android Q Beta***

Terakhir ada *Android Q Beta* yang baru diluncurkan pada 13 Maret 2019 dan saat ini masih terbatas pada beberapa perangkat HP *Android* saja. Seperti pada seri *smartphone Google,* yakni *Google Pixel, Google Pixel XL, Google Pixel 2, Google Pixel 2 XL, Google Pixel 3, Google Pixel 3 XL,* dan *Google Pixel 3 Lite.* Salah satu fitur *Android Q Beta* adalah *Dark Mode* alias mode gelap yang diklim mampu meningkatkan performa baterai. *(Sumber : https: //jalantikus. com/tips urutan-versi-android/ )*

### II.2.3. Basis Data (*Database*)

Basisdata merupakan kumpulan dari data yang saling berhubungan antara satu dengan yang lain. Basis data atau *database* merupakan salah satu komponen yang penting dalam sistem informasi, karena berfungsi berfungsi sebagai basis penyedia informasi bagi pemakainya, Sistem basis data adalah suatu sistem informasi yang mengintegrasikan kumpulan dari data yang saling berhubungan dengan yang lainnya dan untuk membuatnya tersedia beberapa aplikasi yang bermacam-macam dalam suatu sistem organisasi. Sistem basis data adalah suatu sistem menyusun dan mengelola *record-record* menggunakan komputer untuk menyimpan atau merekam serta memelihara data operasional lengkap sebuah organisasi atau perusahaan sehingga mampu menyediakan informasi yang optimal yang diperlukan. (Priyo Sutopo,dkk 2016).

### II.2.4. Normalisasi

Normalisasi adalah suatu teknik untuk mengorganisasikan data kedalam tabel-tabel untuk memenuhi kebutuhan pemakai didalam suatu organisasi. Tujuan dari normalisasi adalah :

a. Untuk menghilangkan kerangkapan data.

b. Untuk mengurangi kompleksitas.

c. Untuk mempermudah pemodifikasian data.

Proses normalisasi antara lain :

a. Data diuraikan dalam bentuk tabel, selajutnya dianalisis berdasarkan persyaratan tertentu kebeberapa tingkat.
b. Apabila tabel yang diuji belum memenuhi persyaratan tertentu maka tabel tersebut perlu dipecah menjdi beberapa tabel yang lebih sederhana sampai memenuhi data yang optimal.

Bentuk-bentuk dari normalisasi adalah :

a. Bentuk tidak normal (*unformalized form*)

   Bentuk ini merupakan bentuk data yang direkam, tidak ada keharusan untuk mengikuti suatu format tertentu, dapat saja data tidak lengkap atau terduplikasi.

b. Bentuk normal pertama (1NF atau *first normal form*)

   Bentuk normal pertama mempunyai ciri-ciri yaitu setiap data dibentuk dalam *flat file* (file dasar) dan data dibentuk dalam satu *record* demi satu record. Tidak ada set atribut yang berulang-ulang atau atribut yang bernilai ganda.

c. Bentuk normal kedua (2NF atau *second normal form*)

Bentuk normal kedua mempunyai syarat yaitu bentuk data telah memenuhi kriteria bentuk normal pertama, atribut bukan kunci haruslah bergantung secara fungsi pada kunci utama, atau *primery key*, sehingga untuk bentuk normal kedua haruslah sudah ditentukan kunci-kunci *field*. Kunci *field* harus unik dan dapat mewakili atribut lain yang menjadi anggotanya.

d. Bentuk normal ketiga (3NF atau *three normal form*)

Untuk menjadi bentuk normal ketiga maka relasi haruslah dalam bentuk normal kedua dan sama atribut bukan primer tidak punya hubungan yang transi, dengan kata lain setiap atribut bukab kunci haruslah bergantun pada *primery key* secara menyeluruh. (Ganda Yoga Swara, 2016).

### II.2.5. XAMPP

XAMPP adalah perangkat lunak bebas, yang mendukung banyak sistem operasi, merupakan kompilasi dari beberapa program. Fungsinya adalah sebagai server yang berdiri sendiri (*localhost*), yang terdiri atas program *Apache HTTP Server*, *MySQL database*, dan penerjemah bahasa yang ditulis dengan bahasa pemrograman *PHP* dan *Perl*. Nama *XAMPP* merupakan singkatan dari X (empat sistem operasi apapun), *Apache*, *MySQL*, *PHP* dan *Perl*. Program ini tersedia dalam *GNU General Public License* dan bebas, merupakan *web server* yang mudah digunakan yang dapat melayani tampilan halaman *web* yang dinamis. Untuk mendapatkanya dapat men-*download* langsung dari *web* resminya. (Randi V Palit, 2015 : 2).

### II.2.6. MySQL

MySQL adalah sebuah implementasi dari sistem manajemen basisdata relasional (RDBMS) yang didistribusikan secara gratis dibawah lisensi GPL (*General Public License*). Setiap pengguna dapat secara bebas menggunakan MySQL, namun dengan batasan perangkat lunak tersebut tidak boleh dijadikan produk turunan yang bersifat komersial. MySQL sebenarnya merupakan turunan salah satu konsep utama dalam basisdata yang telah ada sebelumnya; SQL (*Structured Query Language*). SQL adalah sebuah konsep pengoperasian basisdata, terutama untuk pemilihan atau seleksi dan pemasukan data, yang memungkinkan pengoperasian data dikerjakan dengan mudah secara otomatis.

Kehandalan suatu sistem basisdata (DBMS) dapat diketahui dari cara kerja pengoptimasi-nya dalam melakukan proses perintah-perintah SQL yang dibuat oleh pengguna maupun program-program aplikasi yang memanfaatkannya. Sebagai peladen basis data, *MySQL* mendukung operasi basisdata transaksional maupun operasi basisdata non-transaksional. Pada modus operasi non-transaksional, MySQL dapat dikatakan unggul dalam hal unjuk kerja dibandingkan perangkat lunak peladen basisdata kompetitor lainnya. Namun pada modus non-transaksional tidak ada jaminan atas reliabilitas terhadap data yang tersimpan, karenanya modus non-transaksional hanya cocok untuk jenis aplikasi yang tidak membutuhkan reliabilitas data seperti aplikasi blogging berbasis *web* (*wordpress*), CMS, dan sejenisnya. Untuk kebutuhan sistem yang ditujukan untuk bisnis sangat disarankan untuk menggunakan modus basisdata transaksional,

hanya saja sebagai konsekuensinya unjuk kerja MySQL pada modus transaksional tidak secepat unjuk kerja pada modus non-transaksional.

MySQL pada awalnya diciptakan pada tahun 1979, oleh Michael "Monty" Widenius, seorang programmer komputer asal Swedia. Monty mengembangkan sebuah sistem *database* sederhana yang dinamakan *UNIREG* yang menggunakan koneksi low-level ISAM database engine dengan indexing. Pada saat itu Monty bekerja pada perusahaan bernama TcX di Swedia. TcX pada tahun 1994 mulai mengembangkan aplikasi berbasis web, dan berencana menggunakan *UNIREG* sebagai sistem database. Namun sayangnya, UNIREG dianggagap tidak cocok untuk database yang dinamis seperti *web*. *(*Herpendi : 2016*)*

### II.2.7. PHP

PHP merupakan Bahasa pemograman yang digunakan untuk membuat website dinamis dan interaktif. *Dinamis* artinya, *website* tersebut biasa berubah-ubah tampilan dan kontennya sesuai kondisi tertentu. Sebagai contoh, *PHP* biasa menampilkan tanggal dan hari saat ini secara berganti-ganti didalam sebuah *website*. Interaktif artinya, *PHP* dapat memberi *feedback* bagi *user* (misalnya menampilkan hasil pencarian produk). (Jubile Enterprise : 2018 ; 1).

### II.2.8. Pemesanan/ Reservasi

Reservasi merupakan proses, pembuatan, cara memesan (tempat dan barang) kepada orang lain. Reservasi adalah sebuah proses perjanjian berupa pemesanan sebuah produk baik barang maupun jasa dimana pada saat itu telah

terdapat kesepahaman antara konsumen dengan produsen mengenai produk tersebut namun belum ditutup oleh sebuah transaksi jual-beli. Pada saat reservasi berlangsung biasanya ditandai dengan adanya proses tukar menukar informasi antara konsumen dan produsen agar kesepahaman mengenai produk dapat terwujud. Alasan reservasi menjadi sebuah media yang sangat efektif baik bagi produsen maupun bagi konsumen adalah produsen akan dapat melakukan evaluasi terhadap produk yang akan mereka jual melalui tingkat tinggi rendahnya jumlah reservasi jauh sebelum produk tersebut dijual (barang) ataupun diselenggarakan (jasa), dimana hasil evaluasi tersebut akan membantu produsen untuk menentukan langkah pemasaran yang akan diambil terhadap produk yang akan dijual tersebut. (Roni Ameldi ; 2018).

### II.2.9. Restfull Web Service

REST (*REpresentational State Transfer*) merupakan standar arsitektur komunikasi berbasis web yang sering diterapkan dalam pengembangan layanan berbasis web. Umumnya menggunakan HTTP (*Hypertext Transfer Protocol*) sebagai protocol untuk komunikasi data. REST pertama kali diperkenalkan oleh Roy Fielding pada tahun 2000. Pada arsitektur REST, REST server menyediakan *resources* (sumber daya/data) dan REST client mengakses dan menampilkan *resource* tersebut untuk penggunaan selanjutnya. Setiap *resource* diidentifikasi oleh URIs (*Universal Resource Identifier*s) atau global ID. *Resource* tersebut direpresentasikan dalam bentuk format teks, JSON atau XML. Pada umumnya formatnya menggunakan JSON dan XML.

(https://www.codepolitan.com/mengenal-restful-web-services).

Keuntungan REST :

1. Bahasa dan platform agnostic
2. Lebih sederhana/simpel untuk dikembangkan ketimbang SOAP
3. Mudah dipelajari, tidak bergantung pada tools
4. Ringkas, tidak membutuhkan layer pertukaran pesan (messaging) tambahan
5. Secara desain dan filosofi lebih dekat dengan web

Kelemahan REST :

1. Mengasumsi model point-to-point komunikasi - tidak dapat digunakan untuk lingkungan komputasi terdistribusi di mana pesan akan melalui satu atau lebih perantara
2. Kurangnya dukungan standar untuk keamanan, kebijakan, keandalan pesan, dll, sehingga layanan yang mempunyai persyaratan lebih canggih lebih sulit untuk dikembangkan ("dipecahkan sendiri")
3. Berkaitan dengan model transport HTTP.

(https://www.codepolitan.com/mengenal-restful-web-services).

### II.2.10. *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 :

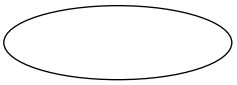
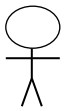
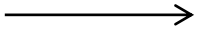
a. Menggambarkan batasan sitem dan fungsi - fungsi sistem secara umum, dibuat dengan *use case* dan *actor*.
b. Menggambarkan kegiatan atau proses bisnis yang dilaksanakan secara umum, dibuat dengan *interaction diagrams*.
c. Menggambarkan representasi struktur *static* sebuah sistem dalam betuk *class 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 *fungsionality* dengan *stereotypes*. (Ommi Alfina dan Fitriana Harahap : 2019)

Pemodelan penggunaan UML merupakan metode pemodelan berorientasi objek dan berbasis visual. Karenanya pemodelan objek yang fokus pada pendefinisian struktur statis dan model sistem informasi yang dinamis daripada mendefinisikan data dan model proses yang tujuannya adalah pengembangan tradisional. UML menawarkan diagram yang dikelompokkan manjadi lima perspektif berbeda untuk memodelkan suatu sistem. Seperti satu set *blue print* yang digunakan untuk membangun sebuah rumah (Saipul Anwar, et al., 2016 : 75-76).

### II.2.10.1. *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.1:

**Tabel II.1. Simbol *Use Case***

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

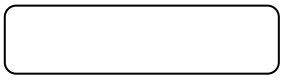
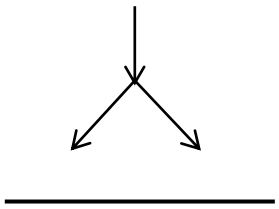
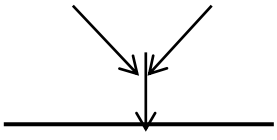
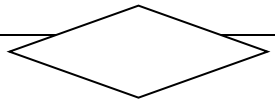
| | | |
|---|---|---|
|  | *Extend* | Merupakan perluasan dari *use case* lain jika kondisi atau syarat terpenuhi. |

**(Sumber : Ade Hendini, 2016)**

### II.2.10.2. *Activity* Diagram

*Activity* diagram menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis. Simbol-simbol yang digunakan dalam *activity diagram* dapat dilihat pada Tabel II.2:

**Tabel II.2. Simbol *Activity Diagram***

| Gambar | Keterangan | Deskripsi |
|---|---|---|
|  | *Start point* | Diletakkan pada pojok kiri atas dan merupakan awal aktifitas. |
|  | *End point* | Akhir aktifitas. |
|  | *Activites* | Menggambarkan suatu proses/kegiatan bisnis. |
|  | *Fork* (Percabangan). | Digunakan untuk menunjukkan kegiatan yang dilakukan secara parallel atau untuk menggabungkan dua kegiatan pararel menjadi satu. |
|  | *Join* (penggabungan) | Digunakan untuk menunjukkan adanya dekomposisi. |
|  | *Decision Points* | Menggambarkan pilihan untuk |

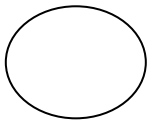
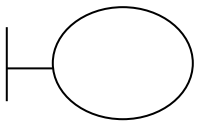
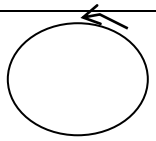
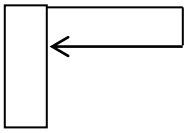
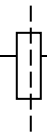
| | | |
|---|---|---|
| | | pengambilan keputusan, *true, false.* |
| New Swimline | *Swimlane* | Pembagian *activity* diagram untuk menunjukkan siapa melakukan apa. |

**(Sumber : Ade Hendini, 2016 : 109)**

### II.2.10.3. *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.3:

**Tabel II.3. Simbol *Sequence Diagram***

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

| | | |
|---|---|---|
| | | berbanding lurus dengan durasi aktivitas sebuah operasi. |
| | *Lifeline* | Garis titik-titik yang terhubung dengan objek, sepanjang *lifeline* terdapat *activation*. |

**(Sumber : Ade Hendini, 2016 : 110)**

### II.2.10.4. *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 *Assosiations*, *Generalitation* dan *Aggregation*, attribut (*Attributes*), operasi (*operation/method*) dan *visibility*, tingkat akses objek eksternal kepada suatu operasi atau attribut. Hubungan antar kelas mempunyai keterangan yang disebut dengan *Multiplicity* atau *Cardinality* (Ade Hendini, 2016 : 111).

**Tabel II.4. *Multiplicity Class Diagram***

| Multiplicity | Penjelasan |
|---|---|
| 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)**