BAB II

TINJAUAN PUSTAKA

II.1. Penelitian Terdahulu

Berdasarkan penelitian yang dilakukan oleh Taufik, dkk (2014) mengenai Rancang Bangun Aplikasi Mobile Untuk Notifikasi Jadwal Kuliah Berbasis Android, Taufik, dkk menyimpulkan bahwa Kehadiran smartphone Android sebagai salah satu produk teknologi terbaru di bidang selular diharapkan dapat membantu mahasiswa mengakses informasi jadwal dengan efektif dan efisien. Kemampuan smartphone Android untuk selalu terkoneksi dengan internet dapat membantu mahasiswa memantau jadwal kuliah secara realtime. Sifat smartphone Android yang mudah dibawa dapat memudahkan mahasiswa untuk mengakses informasi jadwal dimana saja.

Berdasarkan penelitian yang dilakukan oleh Irvan (2017) mengenai Rancang Bangun Sistem Penjadwalan Guru Mengajar Berbasis Web, Zulfauzi menyimpulkan bahwa Untuk menyusun jadwal yang baik, pembuat jadwal perlu memperhatikan bahwa jadwal yang di buat tidak ada bentrokan atau kesamaan jam antar guru satu dengan guru yang lain dalam satu waktu dan kelas tertentu, atau satu guru yang berada di lebih dari satu kelas pada satu waktu tertentu. Begitu pula dengan bentuk penginformasian jadwal masih berjalan secara manual dengan menggunakan media kertas yang di tempel di madding ruang guru.

Berdasarkan beberapa penelitian di atas tidak ditemukan adanya kesamaan penelitian baik dari judul maupun isi. Oleh karena itu, penelitian terdahulu dapat di ambil beberapa referensi untuk penelitian ini.

II.2. Landasan Teori

Berikut ini adalah beberapa referensi yang di kutip dari beberapa sumber jurnal dan buku :

II.2.1. Aplikasi

Menurut Jogiyanto Aplikasi adalah penggunaan dalam suatu komputer, instruksi (*instruction*), atau pernyataan (*statement*) yang disusun sedemikian rupa sehingga komputer dapat memproses *input* menjadi *output*.

Menurut Kamus *Kamus Besar Bahasa Indonesia* (1998:52), aplikasi adalah penerapan dari rancang sistem untuk mengolah data yang menggunakan aturan atau ketentuan bahasa pemrograman tertentu.

Dari pendapat diatas maka dapat disimpulkan Aplikasi adalah Program yang dibuat oleh manusia yang berfungsi untuk menyelesaikan permasalahan-permasalahan masalah yang akan dihadapi. (Zulfauzi, 2015 : 57).

II.2.2. *Client Server*

client-server merupakan sebuah paradigma dalam teknologi informasi yang merujuk kepada cara untuk mendistribusikan aplikasi ke dalam dua pihak : pihak *clien* dan pihak *server*. Dalam model *client*/*server*, sebuah aplikasi dibagi menjadi

dua bagian yang terpisah, tapi masih merupakan sebuah kesatuan yakni komponen klien dan komponen *server*. Komponen *client* juga sering disebut sebagai *front-end*, sementara komponen *server* disebut sebagai *back-end*. Komponen *client*dari aplikasi tersebut dijalankan dalam sebuah workstation dan menerima masukan data dari pengguna. Komponen *client*tersebut akan menyiapkan data yang dimasukkan oleh pengguna dengan menggunakan teknologi pemrosesan tertentu dan mengirimkannya kepada komponen *server* yang dijalankan di atas mesin *server*, umumnya dalam bentuk *request* terhadap beberapa layanan yang dimiliki oleh *server*. Komponen *server* akan menerima *request* dari *client*, dan langsung memprosesnya dan mengembalikan hasil pemrosesan tersebut kepada *client*. *Client*pun menerima informasi hasil pemrosesan data yang dilakukan *server* dan menampilkannya kepada pengguna, dengan menggunakan aplikasi yang berinteraksi dengan pengguna. (Arifin dan Sutariyani, 2014 : 38).

II.2.3. *Android*

Sistem operasi *Android*merupakan sebuah 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. Telepon seluler menggunakan berbagai macam sistem operasi seperti Symbian OS®, Microsoft's Windows Mobile®, Mobile Linux®, iPhone OS® (berdasarkan Mac OS X),

Moblin® dari Intel, dan berbagai macam sistem operasi lainnya. (Sede, dkk, 2015 : 1).

Android adalah sistem operasi disematkan pada *gadget*, baik itu *handphone*, tablet, juga sekarang sudah merambah kekamera digital dan jam tangan. Saat ini gadget berbasis *Android*,baik itu tablet atau *handphone*, begitu digandrung.Selain harganya yang semakin terjangkau,juga banyak varian spesifikasi yang bisa dipilih sesuai kebutuhan dan kantong.

Untuk kebutuhan yang lebih praktis,tablet dan handpone pintar ini bisa menggantikan peran dari sebuah komputer jinjing, terutama untuk kebutuhan *entertaiment*, seperti mendengarkan lagu,menonton *video*,mengirim *email*,bermain *game*, *twitter*, atau *facebook*, juga kegiatan hiburan *online* lainnya**.** (Wahadyo, 2013 : 2).

II.2.4. *Java*

Java adalah bahasa pemrograman yang *multi platform* dan *multi device*. Sekali anda menuliskan sebuah program dengan menggunakan *Java*. Aplikasi dengan berbasis *Java* ini dikompulasikan ke dalam *p-code* dan bisa dijalankan dengan *Java Virtual Machine*. Fungsionalitas dari *Java* ini dapat berjalan dengan *platform* sistem operasi yang berbeda karena sifatnya yang umum dan non spesifik. (Hardianto dan Handaga : 2015 : 4).

Java adalah sebuah bahasa pemrograman yang populer dikalangan para akademisi dan praktisi komputer. *Java* pertama kali dikembangkan untuk memenuhi

kebutuhan akan sebuah bahasa komputer yang ditulis satu kali dan dapat dijalankan dibanyak sistem komputer berbeda tanpa perubahan kode berarti. Pada umumnya, para pakar pemrograman berpendapat bahwa bahasa *Java* memiliki konsep yang konsisten dengan teori pemrograman objek dan aman untuk digunakan. (Wardhani dan Yaqin, 2013 : 474).

II.2.5. *Hypertext Markup Language* (HTML)

Hypertext Markup Language (HTML) adalah suatu bahasa yang dikenali oleh *web browser* untuk menampilkan informasi dengan lebih menarik dibandingkan dengan tulisan teks biasa (*plaint text*). Sedangkan *web browser* adalah bahasa program komputer yang digunakan untuk membaca HTML, kemudian menerjemahkan dan menampilkan hasilnya secara *visual* ke layar komputer. (Nugraha, dkk, 2014 : 175).

II.2.5.1. Struktur Dasar HTML

HTML merupakan dokumen yang terstruktur. HTML setidaknya memiliki struktur dasar yang terdiri dari :

1. *Tag* DTD atau DOCTYPE.
2. *Tag* HTML.
3. *Tag* HEAD.
4. *Tag* BODY. (Nugraha, dkk, 2014 : 175).

II.2.6. *Hypertext Preprocessor* (PHP)

*Hypertext Preprocessor*adalahbahasa skrip yang dapat ditanamkan ataudisisipkan ke dalam HTML. PHP banyakdipakai untuk memrogram situs web dinamis.PHP dapat digunakan untuk membangunsebuah *CMS*.Pada awalnya PHP merupakankependekan dari *Personal Home Page* (Situspersonal). PHP pertama kali dibuat olehRasmus Lerdorf pada tahun 1995. Pada waktuitu PHPmasih bernama *Form Interpreted* (FI),yang wujudnya berupa sekumpulan skrip yangdigunakan untuk mengolah data formulir dari*web*.Selanjutnya Rasmus merilis kodesumber tersebut untuk umum danmenamakannya PHP/FI. Dengan perilisan kodesumber ini menjadi sumber terbuka, makabanyak pemrogram yang tertarik untuk ikutmengembangkan PHP. (Anisya, 2013 : 51).

II.2.7. *Extensible Markup Language* (XML)

Extensible Markup Language (XML) adalah sebuah bahasa markah untuk mendeskripsikan data. XMLmerupakan turunan (*subset*) atau versi ringkas dari SGML (*Standart Generalized Markup Language*), sedangkan SGML merupakan sebuah standar ISOuntuk format dokumen. SGMLtidak berisi berupa *tag-tag* siap pakai seperti halnya bahasa HTML, melainkan berupa aturan-aturan standar dalam pembuatan *tag-tag* format dokumen. SGML banyak dipakai untuk mengelola dokumen dalam jumlah besar, frekuensi revisi tinggi dan dibutuhkan dalam beragam format tampilan. SGML jarang dipakai karena sangat rumit dan kompleks. XML

dibuat dengan konsep yang lebih sederhana dan ringkas, tujuannya agar bisa dipakai sebagai aplikasi *desktop* dan jaringan internet. (Yenni dan Shamir, 2012 : 107).

II.2.8. Basis Data (*Database*)

Database adalah sekumpulan tabel-tabel yang saling berelasi, relasi tersebut bisa ditunjukkan dengan kunci dari tiap tabel yang ada. Satu *database* menunjukkan satu kumpulan data yang dipakai dalam satu lingkup perusahaan atau instansi.*Database* mempunyai kegunaan dalam mengatasi penyusunan dan penyimpanan data, maka seringkali masalah yang dihadapi adalah:

1. Redundansi dan Inkonsistensi data
2. Kesulitan dalam pengaksesan data
3. Isolasi data untuk standarisasi
4. *Multi user*
5. Keamanan data
6. Integritas data
7. Kebebasan data. (Urva dan Siregar, 2015 : 95)

II.2.9. Normalisasi

Normalisasi merupakan parameter digunakan untuk menghindari duplikasiterhadap tabel dalam basis data dan jugamerupakan proses mendekomposisikansebuah tabel yang masih memiliki beberapa anomali atau ketidakwajaran sehinggamenghasilkan tabel yang lebih sederhanadan struktur yang bagus, yaitu sebuah tabelyang tidak memiliki *data redundancy* danmemungkinkan

user untuk melakukan*insert*, *delete*, dan *update* pada baris(*record*) tanpa menyebabkan inkonsistensidata.

1. *First Normal Form (1 NF)*

   Sudah tidak ada *repeating group* yaitupengulangan yang terjadi pada beberapaatribut atau kolom dalam sebuah tabel,dan juga setiap atribut harus bernilaitunggal. Atribut *multivalued,composite, derive* tidak tunggal. Setiapnilai dari atribut hanya mempunyai nilaitunggal.

2. *Second Normal Form (2 NF)*

   Untuk menjadikan tabel normal tingkatke 2 maka sudah 1NF dan setiap atributyang bukan *primary key* sepenuhnyasecara *functional* tergantung padasemua atributpembentuk *primary key*.

3. *Third Normal Form (3 NF)*

   Tabel sudah 2NF dan tidak memiliki*transitive dependencies*, *Transitive dependency* adalah ketika ada atributyang secara tidak langsung tergantungpada *primary key* dan atribut tersebutjuga tergantung pada atribut lain yangbukan *primary key*.

4. *Boyce-codd Normal Form (BCNF)*

   Tabel dalam BCNF jika sudah 3NF dansemua *determinants* adalah *candidatekeys*. Perbedaan 3NF dan BCNF adalahuntuk *functional dependency* A B,3NF memperbolehkan ketergantunganada dalam relasi jika B adalah *PrimaryKey* dan A

bukan merupakan *candidatekey*. Sedangkan BCNF menuntut untukketergantungan tetap ada dalam relasi,A harus menjadi *candidate key*.

5. *Fourth Normal Form* (4 NF)

   Relasi berada pada bentuk normalkeempat apabila memenuhi syaratBCNF dan tidak mempunyai *multivaluedependency.*

6. *Fifth Normal Form* (5 NF)

   Tabel bentuk normal kelima seringdisebut PJNF (*Projection Join NormalForm*), penyebutan PJNF karena untuksuatu relasi akan berbentuk normalkelima jika tabel tersebut dapat dipecahatau diproyeksikan menjadi beberapatabel dan dari

   proyeksi-proyeksi itudapat disusun kembali (*join*) menjaditabel yang sama dengan keadaansemula. Jika penyusunan ini tidakmungkin dilakukan dikatakan padarelasi itu terdapat *join dependencies*dan dikatakan bersifat *lossy join*. (Triyono, 2012 : 19).

II.2.10. MySQL

MySQL adalah *Relation DatabaseManagement System* (RDBMS) yangdidistribusikan secara gratis di bawah lisensiGPL (*General Public License).* MySQLmerupakan turunan dari salah satu konseputama dalam *database* sejak lama, yaitu SQL(*Structure Query Language)*. SQL merupakansalah satu konsep pengoperasian *database,*terutama sebagai seleksi dan pemasukan data,yang memungkinkan pengoperasian datanyadikerjakan dengan mudah secara otomatis. (Inayah, dkk, 2015 : 5).

II.2.10.1. Perintah Dasar MySQL

Perintah dasar MySQL dalam pengolahan *database* adalah sebagai berikut :

1. *Insert*(Berfungsi untuk memasukkan data baru kedalam tabel *database*)
2. *Update*(Berfungsi untuk merubah isi data tertentu di dalam tabel *database*)
3. *Delete*(Berfungsi untuk menghapus isi data tertentu di dalam tabel *database*). (Inayah, dkk, 2015 : 5).

II.2.11. XAMPP

XAMPP adalah perangkat lunak bebas (*free software*), yang mendukung untuk banyak sistem operasi dan merupakan kompilasi dari beberapa program. Fungsi XAMPP sendiri sebagai *server* yang berdiri sendiri (*localhost*), yang terdiri dari beberapa program, antara lain *Apache* HTTP *Server*, MySQL *Database* dan penerjemah bahasa yang di tulis dengan bahasa pemrograman PHP dan Perl. (Haqi, 2017 : 7).

II.2.12. *Unified Modeling Language* (UML)

UML merupakan metodologi dalam mengembangkan sistem berorientasi objek dan juga merupakan alat untuk mendukung pengembangan sistem. UML saat ini sangat banyak dipergunakan dalam dunia industri yang merupakan standar bahasa

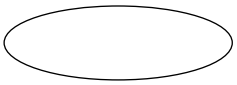
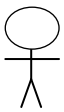
pemodelan umum dalam industri perangkat lunak dan pengembangan sistem. (Urva dan Siregar, 2015 : 93).

Alat bantu yang digunakan dalam perancangan berorientasi objek berbasiskan UML adalah sebagai berikut:

1. *Use Case* Diagram

Use case diagram merupakan pemodelan untuk kelakukan (*behavior*) sistem informasi yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. Dapat dikatakan *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 dibawah ini:

Tabel II.1. Simbol *Use Case*

Gambar	Keterangan
	*Use case* menggambarkan fungsionalitas yang disediakan sistem sebagai unit-unit yang bertukar pesan antar unit dengan aktor, dan dinyatakan dengan menggunakan kata kerja di awal nama *use case*.
	Aktor adalah *abstraction* dari orang atau sistem yang lain yang mengaktifkan fungsi dari target sistem. Untuk mengidentifikasikan aktor, harus ditentukan pembagian tenaga kerja dan tugas-tugas yang

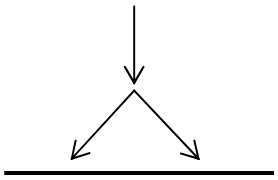
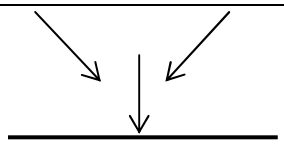
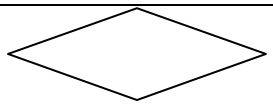
	berkaitan dengan peran pada konteks target sistem. Orang atau sistem bisa muncul dalam beberapa peran. Perlu dicatat bahwa aktor berinteraksi dengan *use case*, tetapi tidak memiliki *control* terhadap *use case*.
	Asosiasi antara aktor dan *use case*, digambarkan dengan garis tanpa panah yang mengindikasikan siapa atau apa yang meminta interaksi secara langsung dan bukannya mengidikasikan aliran data.
	Asosiasi 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:Urva dan Siregar, 2015 : 94)

2. Diagram Aktivitas (*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.3 dibawah ini:

Tabel II.2. Simbol *Activity Diagram*

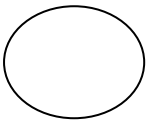
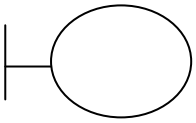
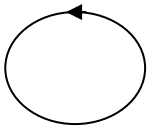
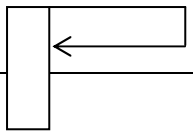
Gambar	Keterangan
	*Start point*, diletakkan pada pojok kiri atas dan merupakan awal aktifitas.
	*End point*, akhir aktifitas.
	*Activites*, menggambarkan suatu proses/kegiatan bisnis.
	*Fork* (Percabangan), digunakan untuk menunjukkan kegiatan yang dilakukan secara parallel atau untuk menggabungkan dua kegiatan pararel menjadi satu.
	*Join* (penggabungan) atau rake, digunakan untuk menunjukkan adanya dekomposisi.
	*Decision Points,* menggambarkan pilihan untuk pengambilan keputusan, *true, false.*
New Swimline	*Swimlane*, untuk menunjukkan siapa melakukan apa.

(Sumber : Urva dan Siregar, 2015 : 94)

3. Diagram Urutan (*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 dibawah ini :

Tabel II.3. Simbol *Sequence Diagram*

Gambar	Keterangan
	*EntityClass,* 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, 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 : Urva dan Siregar, 2015 : 95)

4. *Class Diagram* (Diagram Kelas)

Merupakan hubungan antar kelas dan penjelasan detail tiap-tiap kelas di dalam model desain dari suatu sistem, juga memperlihatkan aturan-aturan dan tanggng 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 (*Operations/Method*), *Visibility,* tingkat akses objek eksternal kepada suatu operasi atau atribut. Hubungan antar kelas mempunyai keterangan yang disebut dengan *multiplicity* atau kardinaliti yang dapat dilihat pada tabel II.5 dibawah ini:

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 : Urva dan Siregar, 2015 : 95)