BAB II
TEORI PENDUKUNG

2.1 Penelitian Sebelumnya

Di dalam pembuatan aplikasi ini, penulis telah memilih beberapa penelitian-penelitian terdahulu yang dijadikan sebagai referensi. Penelitian yang dilakukan oleh orang lain menghasilkan berbagai hasil yang berbeda-beda. Berikut adalah Penelitian terdahulu yang menjadi acuan peneliti yang sesuai dengan penelitian saat ini antara lain :

Tabel 2.1 Penelitian Sebelumnya

Penulis	Judul Jurnal	Tahun	Penjelasan Isi Jurnal	Perbedaan Penelitian
Arie Setya Putra dan Ochi Marshella Febriani	Sistem Informasi Monitoring Inventori Barang pada Balai Riset Standardisasi Industri Bandar Lampung	2013	Penelitian yang dilakukan bertujuan untuk membantu Balai Riset Standardisasi dan Industri Bandar Lampung dalam melakukan evaluasi data yang baik dengan sistem informasi monitoring inventori barang guna mengontrol peminjaman dan pengembalian barang melalui monitoring inventori	1.Pada penelitian yang dilakukan oleh Arie Setya Putra dan Ochi Marshella Febriani, pembahasan yang dilakukan bukan memonitoring buku melinkan memonitoring inventori barang, sedangkan pada penelitian saat ini pembahasan yang di lakukan untuk memonitoring buku pada gudang. 2. Pada penelitian yang dilakukan oleh Arie Setya Putra dan Ochi Marshella Febriani, pembangunan dan pembuatan perangkat menggunakan metode penelitian analis dan desain terstruktur, sedangkan pada penelitian saat ini menggunakan metode Mix Methode dan waterfall. 3. Pada penelitian yang dilakukan oleh Arie Setya Putra dan Ochi Marshella Febriani, perancangan basis data menggunakan Entity-Relationship Diagram (ERD), sedangkan pada penelitian saat ini menggunakan perancangan basis data Class Diagram. 4. Pada penelitian saat ini pembangunan website menggunakan framework Codeigniter dan berbasis android, sedangkan penelitian sebelumnya tidak menggunakan aplikasi berbasis dekstop.
Tryawan Hendra	Rancang Bangun Aplikasi Monitoring	2016	Penelitian yang dilakukan bertujuan untuk monitoring jamaah haji	1.Pada penelitian yang dilakukan oleh Tryawan Hendra Septian, pembahasan

Penulis	Judul Jurnal	Tahun	Penjelasan Isi Jurnal	Perbedaan Penelitian
Septian	Jamaah Haji Berbasis Mobile Android		berbasis smartphone Android. Aplikasi monitoring ini nantinya akan berjalan di smartphone android dengan mendeteksi lokasi dari jamaah yang kemudian akan memberikan sebuah gambaran tentang lokasi dari jamaah tersebut sedang berada. Aplikasi ini juga akan memberikan informasi jalur perjalanan dari jamaah	yang dilakukan bukan memonitoring buku melinkan memonitoring jamaah haji berbasis android, sedangkan pada penelitian saat ini pembahasan yang di lakukan untuk memonitoring buku pada gudang. 2. Pada penelitian yang dilakukan oleh Tryawan Hendra Septian, pembangunan dan pembuatan perangkat menggunakan metode kualitatif dan waterfall, sedangkan pada penelitian saat ini menggunakan metode Mix Methode dan waterfall. 3. Pada penelitian saat ini pembangunan website menggunakan framework Codeigniter dan berbasis android, sedangkan penelitian sebelumnya tidak menggunakan aplikasi berbasis dekstop.
M. Irwan Hidayat	Aplikasi Monitoring Aktivitas Santri Berbasis Android (Studi Kasus : Pesantren Modern Ulul Al-Bab Makassar)	2016	Aplikasi bertujuan untuk dapat membantu orang tua dalam mengawasi perkembangan anaknya dalam pesantren secara real time.Aplikasi ini merupakan bentuk tanggungjawab dari pihak pesantren kepada orang tua santri agar segala sesuatu yang terjadi dalam pesantren tidak semerta-merta disalahkan kepada pihak pesantren karena dengan monitoring ini pihak orang tua dapat mengontrol nilai dan absensi anaknya dari jauh.	1.Pada penelitian yang dilakukan oleh M. Irwan Hidayat, pembahasan yang dilakukan bukan memonitoring buku melinkan memonitoring aktivitas santri berbasis android, sedangkan pada penelitian saat ini pembahasan yang di lakukan untuk memonitoring buku pada gudang. 2. Pada penelitian yang dilakukan oleh M. Irwan Hidayat, pembangunan dan pembuatan perangkat menggunakan metode deskriptif kualitatif dan waterfall, sedangkan pada penelitian saat ini menggunakan metode Mix Methode dan waterfall. 3. Pada penelitian saat ini pembangunan website menggunakan framework Codeigniter dan berbasis android, sedangkan penelitian sebelumnya tidak menggunakan framework codeigniter.
Susmini Indriani Lestariningati dan Fathur Rozak	Pembangunan Aplikasi Monitoring Jaringan Berbasis Web Menggunakan Simple Network Management Protocol (Snmp)	2015	Penelitian bertujuan memudahkan administrator jaringan didalam melakukan monitoring dan menjaga availability atau ketersediaan layanan di dalam jaringan tersebut. Dari hasil pengujian Alpha yang dilakukan dengan metoda Black Box dihasilkan bahwa aplikasi yang dibangun telah memenuhi fungsi-fungsi yang diharapkan. Dengan	1.Pada penelitian yang dilakukan oleh Susmini Indriani Lestariningati dan Fathur Rozak, pembahasan yang dilakukan bukan memonitoring buku melinkan memonitoring jaringan, sedangkan pada penelitian saat ini pembahasan yang di lakukan untuk memonitoring buku pada gudang. 2. Pada penelitian yang dilakukan oleh Susmini Indriani Lestariningati dan

Penulis	Judul Jurnal	Tahun	Penjelasan Isi Jurnal	Perbedaan Penelitian
			menggunakan aplikasi berbasis web ini dapat memudahkan bagi seorang admin jaringan untuk melakukan monitoring terhadap agent yang tersebar diseluruh jaringan sehingga dapat menjaga ketersediaan atau availability dari jaringan tersebut.	Fathur Rozak, menggunakan model OSI, sedangkan pada penelitian saat ini menggunakan metode Mix Methode dan waterfall. 3. Pada penelitian yang dilakukan oleh Susmini Indriani Lestariningati dan Fathur Rozak, perancangan perancangan mengunakan DFD, sedangkan pada penelitian saat ini menggunakan UML. 4. Pada penelitian saat ini pembangunan website menggunakan framework Codeigniter dan berbasis android, sedangkan penelitian sebelumnya menggunakan aplikasi berbasis desktop.
Nora Lizarti dan Wirta Agustin	Aplikasi Network Traffic Monitoring Menggunakan Simple Network Management Protocol (SNMP) pada Jaringan Virtual Private Network (VPN)	2015	Aplikasi Sistem Informasi yang ada di SMK Labor Binaan FKIP UNRI telah dapat menunjang proses keputusan yang akurat. Aplikasi Sistem Informasi tersebut bisa di akses melalui internet secara private dengan teknologi VPN (Virtual Private Network), tetapi penggunaannya belum bisa di pantau secara spesifik oleh administrator jaringan, sehiingga perlu untuk membangunkan suatu sistem traffic monitoring dengan menggunakan Simple Network Management Protocol (SNMP) dalam mengimplementasikan aplikasi untuk melakukan pengamatan penggunaan lalu lintas data internet dari parameter yang ada pada jaringan VPN (Virtual Private Network) berbasis web, yang menjadi salah satu solusi dalam menyelesaikan masalah yang ada.	1.Pada penelitian yang dilakukan oleh Nora Lizarti dan Wirta Agustin, pembahasan yang dilakukan bukan memonitoring buku melinkan memonitoring Network Traffic, sedangkan pada penelitian saat ini pembahasan yang di lakukan untuk memonitoring buku pada gudang. 2. Pada penelitian saat ini pembangunan website menggunakan framework Codeigniter dan berbasis android, sedangkan penelitian sebelumnya menggunakan aplikasi berbasis desktop.

2.1 Pengertian Aplikasi

Perangkat lunak aplikasi yaitu perangkat lunak yang digunakan untuk membantu pemakai komputer untuk melaksanakan pekerjaannya. Jika ingin mengembangkan program aplikasi sendiri, maka untuk menulis program aplikasi tersebut, dibutuhkan suatu bahasa pemrograman, yaitu language software, yang dapat

berbentuk assembler, compiler ataupun interpreter. Jadi language software merupakan bahasanya dan program yang ditulis merupakan program aplikasinya. Language software berfungsi agar dapat menulis program dengan bahasa yang lebih mudah, dan akan menterjemahkannya ke dalam bahasa mesin supaya bisa dimengerti oleh komputer. Bila hendak mengembangkan suatu program aplikasi untuk memecahkan permasalahan yang besar dan rumit, maka supaya program aplikasi tersebut dapat berhasil dengan baik, maka dibutuhkan prosedur dan perencanaan yang baik dalam mengembangkannya. Sekarang, banyak sekali program-program aplikasi yang tersedia dalam bentuk paket-paket program. Ini adalah program-program aplikasi yang sudah ditulis oleh orang lain atau perusahaan-perusahaan perangkat lunak. Beberapa perusahaan perangkat lunak telah memproduksi paket-paket perangkat lunak yang mempunyai reputasi internasional. Program-program paket tersebut dapat diandalkan, dapat memenhi kebutuhan pemakai, dirancang dengan baik, relatip bebas dari kesalahankesalahan, user friendly (mudah digunakan), mempunyai dokumentasi manual yang memadai, mampu dikembangkan untuk kebutuhan mendatang, dan didukung perkembangannya. Akan tetapi, bila permasalahannya bersifat khusus dan unik, sehingga tidak ada paket-paket program yang sesuai untuk digunakan, maka dengan terpaksa harus mengembangkan program aplikasi itu sendiri. [1]

2.2 Basis Data

Basis data terdiri atas 2 kata, yaitu Basis dan Data. Basis kurang lebih dapat diartikan sebagai markas atau gudang, tempat bersarang/berkumpul. Sedangkan data adalah representasi fakta dunia nyata yang mewakili suatu objek seperti manusia (pegawai, siswa, pembeli, pelanggan), barang, hewan, peristiwa, konsep, keadaan, dan sebagainya, yang direkam dalam bentuk angka, huruf, symbol, teks, gambar, bunyi, atau kombinasinya. Basis data sendiri dapat didefinisiikan dalam sejumlah sudut pandang seperti:

a. Himpunan kelompok data (arsip) yang saling berhubungan yang diorganisasi sedemikian rupa agar kelak dapat dimanfaatkan kembali dengan cepat dan mudah.
b. Kumpulan data yang saling berhubungan yang disimpan secara bersama sedemikian rupa dan tanpa pengulangan (redudansi) yang tidak perlu, untuk memenuhi berbagai kebutuhan.
c. Kumpulan file/tabel/arsip yang saling berhubungan yang disimpan dalam media penyimjpanan elektronis.

Basis data dan lemari arsip sesungguhnya memiliki prinsip kerja dan tujuan yang sama. Prinsip utamanya adalah pengaturan data/arsip. Dan tujuan utamanya adalah kemudahan dan kecepatan dalam pengambilan kembali data/arsip. Perbedaannya hanya terletak pada media penyimpanan yang digunakan. Jika lemari arsip menggunakan lemari dari besi atau kayu sebagai media penyimpanan, maka basis data menggunakan media penyimpanan elektronis seperti disk (disket atau hard disk). Hal ini merupakan konsekuensi yang logis, karena lemari arsip langsung dikelola/ditangani oleh manusia, sementara basis data dikelola/ditangani melalui perantara alat/mesin pintar elektronis (yang dikenal sebagai komputer). Perbedaan media ini yang selanjutnya melahirkan perbedaan media ini yang selanjutnya melahirkan perbedaan-perbedaan lain yang menyangkut jumlah dan jenis metoda/cara yang dapat digunakan dalam upaya penyimpanan.[2]

2.3 Sistem Monitoring

Monitoring didefinisikan sebagai siklus kegiatan yang mencakup pengumpulan, peninjauan ulang, pelaporan, dan tindakan atas informasi suatu proses yang sedang diimplementasikan. Umumnya, monitoring digunakan dalam checking antara kinerja dan target yang telah ditentukan. [5]

Monitoring ditinjau dari hubungan terhadap manajemen kinerja adalah proses terintegrasi untuk memastikan bahwa proses berjalan sesuai rencana (on the track). Monitoring dapat memberikan informasi keberlangsungan proses untuk menetapkan langkah menuju ke arah perbaikan yang berkesinambungan. Pada pelaksanaannya,

monitoring dilakukan ketika suatu proses sedang berlangsung. Level kajian sistem monitoring mengacu pada kegiatan per kegiatan dalam suatu bagian, misalnya kegiatan pemesanan barang pada supplier oleh bagian purchasing. Indikator yang menjadi acuan monitoring adalah output per proses / per kegiatan. [5]

Umumnya, pelaku monitoring merupakan pihak-pihak yang berkepentingan dalam proses, baik pelaku proses (*self monitoring*) maupun atasan / supervisor pekerja. Berbagai macam alat bantu yang digunakan dalam pelaksanaan sistem monitoring, baik observasi / interview secara langsung, dokumentasi maupun aplikasi visual. [5]

Pada dasarnya, monitoring memiliki dua fungsi dasar yang berhubungan, yaitu compliance monitoring dan performance monitoring. Compliance monitoring berfungsi untuk memastikan proses sesuai dengan harapan / rencana. Sedangkan, performance monitoring berfungsi untuk mengetahui perkembangan organisasi dalam pencapaian target yang diharapkan. [5]

Umumnya, output monitoring berupa progress report proses. Output tersebut diukur secara deskriptif maupun non-deskriptif. Output monitoring bertujuan untuk mengetahui kesesuaian proses telah berjalan. Output monitoring berguna pada perbaikan mekanisme proses / kegiatan di mana monitoring dilakukan.[5]

2.4 Efektifitas Sistem Monitoring

Sistem *monitoring* akan memberikan dampak yang baik bila dirancang dan dilakukan secara efektif. Berikut kriteria sistem *monitoring* yang efektif: [5]

1. Sederhana dan mudah dimengerti (*user friendly*). *Monitoring* harus dirancang dengan sederhana namun tepat sasaran. Konsep yang digunakan adalah singkat, jelas, dan padat. Singkat berarti sederhana, jelas berarti mudah dimengerti, dan padat berarti bermakna (berbobot).
2. Fokus pada beberapa indikator utama. Indikator diartikan sebagai titik kritis dari suatu *scope* tertentu. Banyaknya indikator membuat pelaku dan obyek *monitoring* tidak fokus. Hal ini berdampak pada pelaksanaan sistem tidak

terarah. Maka itu, fokus diarahkan pada indikator utama yang benar-benar mewakili bagian yang dipantau.

3. Perencanaan matang terhadap aspek-aspek teknis. Tujuan perancangan sistem adalah aplikasi teknis yang terarah dan terstruktur. Maka itu, perencanaan aspek teknis terkait harus dipersiapkan secara matang. Aspek teknis dapat menggunakan pedoman 5W1H, meliputi apa, mengapa, siapa, kapan, di mana dan bagaimana pelaksanaan sistem *monitoring*.
4. Prosedur pengumpulan dan penggalian data. Selain itu, data yang didapatkan dalam pelaksanaan *monitoring* pada *on going process* harus memiliki prosedur tepat dan sesuai. Hal ini ditujukan untuk kemudahan pelaksanaan proses masuk dan keluarnya data. Prosedur yang tepat akan menghindari proses *input* dan *output* data yang salah (tidak akurat).

2.5 Tujuan Sistem Monitoring

Terdapat beberapa tujuan sistem *monitoring*. Tujuan sistem *monitoring* dapat ditinjau dari beberapa segi, misalnya segi obyek dan subyek yang dipantau, serta hasil dari proses *monitoring* itu sendiri. Adapun beberapa tujuan dari sistem *monitoring* yaitu (Amsler, dkk, 2009) yaitu: [5]

1. Memastikan suatu proses dilakukan sesuai prosedur yang berlaku. Sehingga, proses berjalan sesuai jalur yang disediakan (*on the track*).
2. Menyediakan probabilitas tinggi akan keakuratan data bagi pelaku *monitoring.*
3. Mengidentifikasi hasil yang tidak diinginkan pada suatu proses dengan cepat (tanpa menunggu proses selesai).
4. Menumbuh kembangkan motivasi dan kebiasaan positif pekerja.

2.6 Bentuk-Bentuk Sistem Monitoring

Sistem *monitoring* dapat dilakukan dengan berbagai bentuk/metode implementasi. Bentuk implementasi sistem *monitoring* tidak memiliki acuan baku, sehingga pelaksanaan sistem mengacu ke arah improvisasi individu dengan penggabungan beberapa bentuk. Penggunaan bentuk sistem *monitoring* disesuaikan dengan situasi dan kondisi organisasi. Situasi dan kondisi dapat berupa tujuan organisasi, ukuran

dan sifat proses bisnis perusahaan, serta budaya/etos kerja. Mengemukakan tujuh bentuk aktivitas dari sistem *monitoring*, yaitu (Williams, 1998): [5]

1. Observasi proses kerja, misalnya dengan melakukan visit pada fasilitas kerja, pemantauan kantor, lantai produksi, maupun karyawan yang sedang bekerja
2. Membaca dokumentasi laporan, berupa ringkasan kinerja dan *progress report*
3. Melihat *display* data kinerja lewat layar komputer
4. Melakukan inspeksi sampel kualitas dari suatu proses kerja
5. Melakukan rapat pembahasan perkembangan secara individual maupun grup
6. Melakukan survei klien/konsumen untuk menilai kepuasan akan produk atau layanan jasa suatu organisasi
7. Melakukan survei pasar untuk menilai kebutuhan konsumen sebagai pedoman dalam tindak lanjut perbaikan.

2.7 *Tools* yang di Gunakan

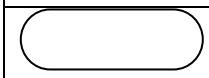
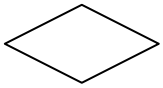
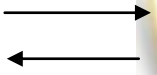
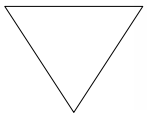
Untuk memudahkan dalam menganalisis dan merancang suatu sistem informasi dibutuhkan *tools* pembantu untuk memodelkannya. Dalam kegiatan Tugas Akhir terdapat dua tahapan yang akan ditangani yaitu menganalisis sistem yang sedang berlangsung dan dan merancang sistem informasi yang digunakan untuk memperbaiki sistem yang sedang berjalan. Pada sub bab ini akan dijelaskan mengenai *tools* pemodelan yang digunakan untuk analisis dan perancangan penilaian kinerja karyawan.

2.7.1 *Flowmap*

Flowmap adalah campuran peta dan *flowchart*, yang menunjukkanpergerakan benda dari satu lokasi ke lokasi lain, seperti jumlah orang dalam migrasi, jumlah barang yang diperdagangkan, atau jumlah paket dalam jaringan. *Flowmap* menolong analisis dan *programmer* untuk memecahkan masalah ke dalam segmen-segmen yang lebih kecil dan menolong dalam menganalisis alternatif-alternatif lain dalam pengoperasian. Simbol-simbol dari *flowmap* adalah sebagai berikut:[6]

Tabel 2.2 Simbol *Flowmap* [8]

Simbol *Flowmap*	Keterangan
	Digunakan untuk menunjukan awal dan akhir program.

Simbol *Flowmap*	Keterangan
 *Terminator*	
 *Process*	Digunakan untuk melakukan pemrosesan.
 *Decision*	Digunakan untuk mewakili operasi perbandingan logika.
Manual Input	Digunakan untuk pemasukan data secara manual *on-line keyboard*.
 *Document*	Digunakan untuk melakukan pembuatan data berupa dokumen.
 *Flow Lines*	Digunakan untuk menunjukkan arus dari proses.
 *Offline Storage*	Digunakan untuk menyimpan data ke dalam suatu media tertentu.
 *Disk and On-line Storage*	Digunakan untuk menyatakan input berasal dari *disk* atau output disimpan ke *disk*.

2.7.2 Metode Black Box Testing

Pengujian blackbox (blackbox testing) adalah salah satu metode pengujian perangkat lunak yang berfokus pada sisi fungsionalitas, khususnya pada input dan

output aplikasi (apakah sudah sesuai dengan apa yang diharapkan atau belum). Tahap pengujian merupakan salah satu tahap yang harus ada dalam sebuah siklus pengembangan perangkat lunak. Black Box Testing merupakan pengujian yang berfokus pada spesifikasi fungsional dari perangkat lunak, tester dapat mendefinisikan kumpulan kondisi input dan melakukan pengetesan pada spesifikasi fungsional program. mengemukakan ciri-ciri black box testing, yaitu: 1. Black box testing berfokus pada kebutuhan fungsional pada software, berdasarkan pada spesifikasi kebutuhan dari software. 2. Black box testing bukan teknik alternatif daripada white box testing. Lebih dari pada itu, ia merupakan pendekatan pelengkap dalam mencakup error dengan kelas yang berbeda dari metode white box testing. Black box testing melakukan pengujian tanpa pengetahuan detil struktur internal dari sistem atau komponen yang dites. juga disebut sebagai behavioral testing, specification-based testing, input/output testing atau functional testing. [7]

2.7.3 UML

UML (*Unified Modeling Language*) adalah sebuah bahasa untuk menentukan visualisasi, kontruksi, dan mendokumentasikan *artifact* (bagian dari informasi yang digunakan atau dihasilkan dalam suatu proses pembuatan perangkat lunak). *Artifact* dapat berupa model, deskripsi, atau perangkat lunak dari sistem perangkat lunak, seperti pada pemodelan bisnis dan sistem non perangkat lunak lainnya. *UML* merupakan suatu kumpulan teknik terbaik yang telah terbukti sukses dalam memodelkan sistem yang besar dan kompleks. *UML* tidak hanya digunakan dalam proses pemodelan perangkat lunak, namun hampir dalam semua bidang yang membutuhkan pemodelan.

UML menyediakan 10 macam diagram untuk memodelkan aplikasi berorientasi objek, yaitu: [8]

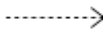
1. *Use Case Diagram* untuk memodelkan proses bisnis.
2. *Conceptual Diagram* untuk memodelkan konsep-konsep yang ada di dalam aplikasi.

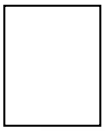
3. *Sequence Diagram* untuk memodelkan pengiriman pesan (*message*) antar *objects*.
4. *Collaboration Diagram* untuk memodelkan interaksi antar *objects*.
5. *State Diagram* untuk memodelkan perilaku *objects* di dalam sistem.
6. *Activity Diagram* untuk memodelkan perilaku *use case* dan *objects* di dalam *system*.
7. *Class Diagram* untuk memodelkan struktur kelas.
8. *Object Diagram* untuk memodelkan struktur *object*.
9. *Component Diagram* untuk memodelkan komponen *object*.
10. *Deployment Diagram* untuk memodelkan distribusi aplikasi.

2.7.3.1 *Use Case* Diagram

Use case diagram menggambarkan fungsionalitas yang diharapkan dari sebuah sistem. Diagram ini menekankan pada "apa" yang diperbuat sistem, dan bukan "bagaimana". Sebuah *use case* merepresentasikan sebuah interaksi antara aktor dengan sistem.

Tabel 2.3 Simbol *Use Case* Diagram [10]

No	Gambar	Nama	Keterangan
1.		*Actor*	Orang atau divisi yang terlibat di dalam suatu sistem.
2.		*Dependency*	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri *(independent)* akan mempengaruhi elemen yang bergantung padanya elemen yang tidak mandiri (*independent*).
3.		*Generalization*	Hubungan dimana objek anak (*descendent*) berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (*ancestor*).
4.		*Include*	Menspesifikasikan bahwa *use case* sumber secara *eksplisit*.

No	Gambar	Nama	Keterangan
5.		*Extend*	Menspesifikasikan bahwa *use case* target memperluas perilaku dari *use case* sumber pada suatu titik yang diberikan.
6.		*Association*	Apa yang menghubungkan antara objek satu dengan objek lainnya.
7.		*System*	Menspesifikasikan paket yang menampilkan sistem secara terbatas.
8.		*Use Case*	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu *actor*.

2.7.3.2 Sistem *Sequence* Diagram dan *Sequence* Diagram

Sequence diagram merupakan sebuah diagram yang menggambarkan interaksi antar objek di dalam sebuah sistem. Interaksi tersebut berupa *message* yang digambarkan terhadap waktu. *Sequence* diagram terdiri dari dimensi horizontal (objek-objek) dan dimensi vertikal (waktu).

Sequence diagram biasa digunakan untuk menggambarkan skenario atau rangkaian langkah-langkah yang dilakukan sebagai respon dari sebuah *event* untuk menghasilkan keluaran tertentu. Diawali dari apa yang memicu aktivitas tersebut, proses dan perubahan apa saja yang terjadi secara internal dan *output* apa yang dihasilkan. Namun, diagram ini kurang mampu menjelaskan *detail* dari sebuah algoritma, seperti *loop*, *branching*.

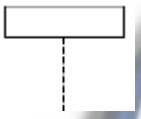
Istilah lain pada *sequence* diagram adalah sebagai berikut:

1. *Time* merupakan elemen penting dalam *sequence* diagram, dan disini konteksnya adalah urutan, bukan durasi.
2. *Alternatives* biasa digunakan untuk mendesain sebuah pilihan *mutually exclusive* antara dua atau lebih urutan pesan. *Alternative* menggambarkan pemodelan dari sebuah *logic* "*if then else*" klasik.

Kombinasi fragmen *option* biasa digunakan untuk memodelkan sebuah urutan dimana sebuah kondisi tertentu akan terjadi, jika tidak-urutan, tidak akan terjadi. Sebuah *option* digunakan untuk memodelkan sebuah *statement* "*if then*" sederhana.

Kombinasi fragmen *loop* digunakan untuk memodelkan sebuah urutan yang berulang-ulang. Dimana syarat perulangan diletakkan pada bagian sebelah kiri atas sebuah fragmen *loop*.

Tabel 2.4 Simbol *Sequence* Diagram [10]

No	Gambar	Nama	Keterangan
1.		*LifeLine*	Objek *entity*, antarmuka yang saling berinteraksi.
2.		*Message*	Spesifikasi dari komunikasi antar objek yang memuat informasi-informasi tentang aktivitas yang terjadi.

2.7.3.3 *Conceptual Class* Diagram

Conceptual Class Diagram menggambarkan identifikasi kelas konseptual dengan atribut dan asosiasinya. *Conceptual Class* Diagram bertujuan untuk menjabarkan domain permasalahan menjadi domain model.

2.7.3.4 *Class* Diagram

Class adalah sebuah spesifikasi yang jika diinstansiasi akan menghasilkan sebuah objek dan merupakan inti dari pengembangan dan desain berorientasi objek. *Class* menggambarkan keadaan (atribut/properti) suatu sistem, sekaligus menawarkan layanan untuk memanipulasi keadaan tersebut (metoda/fungsi).

Class diagram menggambarkan struktur dan deskripsi *class*, *package*, dan objek beserta hubungan satu sama lain seperti pewarisan, asosiasi, dan lain-lain.

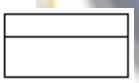
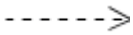
Class memiliki tiga area pokok:

1. Nama
2. Atribut
3. Metoda

Atribut dan metoda dapat memiliki salah satu sifat berikut:

a. *Private*, tidak dapat dipanggil dari luar *class* yang bersangkutan.
b. *Protected*, hanya dapat dipanggil oleh *class* yang bersangkutan dan anak-anak yang mewarisinya.
c. *Public*, dapat dipanggil oleh siapa saja.

Tabel 2.5 Simbol *Class* Diagram [10]

No	Gambar	Nama	Keterangan
1.		*Generalization*	Hubungan dimana objek anak *(descendent)* berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (*ancestor*).
2.		*Nary Association*	Upaya untuk menghindari asosiasi dengan lebih dari 2 objek.
3.		*Class*	Himpunan dari objek-objek yang berbagi atribut serta operasi yang sama.
4.		*Collaboration*	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu aktor.
5.		*Realization*	Operasi yang benar-benar dilakukan oleh suatu objek.
6.		*Dependency*	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri *(independent)* akan mempengaruhi elemen yang bergantung padanya, elemen yang tidak mandiri.
7.		*Association*	Apa yang menghubungkan antara objek satu dengan objek lainnya.

Cardinalitas atau *multiplicity* adalah simbol yang menunjukan jumlah banyaknya objek sebuah *class* yang berelasi dengan sebuah objek lain pada *class* lain yang berasosiasi dengan *class* tersebut.

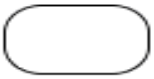
Tabel 2.6 Cardinalitas/Multiplicity Pada Class Diagram [10]

No	Cardinalitas/Multiplicity	Keterangan
1.	*	Banyak
2.	0	Tepat Nol
3.	1	Tepat Satu
4.	0..*	Minimal Nol, Maksimal Tak Terhingga
5.	1..*	Minimal Satu, Maksimal Tak Terhingga
6.	0..1	Minimal Nol, Maksimal Satu

2.7.3.5 *Activity* Diagram

Activity diagram adalah representasi grafis dari alur kerja tahapan aktivitas. Diagram ini mendukung pilihan tindakan, iterasi, dan *concurrency*. Pada pemodelan *UML*, *activity diagram* dapat digunakan untuk menjelaskan bisnis dan alur kerja professional/secara *step-by-step* dari komponen suatu sistem. *Activity diagram* menujukkan keseluruhan dari aliran control. [8]

Tabel 2.7 Simbol Activity Diagram [10]

No	Gambar	Nama	Keterangan
1.		*Activity*	Memperlihatkan bagaimana masing-masing kelas antarmuka 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.

No	Gambar	Nama	Keterangan
5.		*Fork Node*	Satu aliran yang pada tahap tertentu berubah menjadi beberapa aliran.

2.7.4 PHP

PHP adalah singkatan dari "PHP: Hypertext Prepocessor", yaitu bahasa pemrograman yang digunakan secara luas untuk penanganan pembuatan dan pengembangan sebuah *situs web* dan bisa digunakan bersamaan dengan *HTML*. *PHP* diciptakan oleh Rasmus Lerdorf pertama kali tahun 1994. Pada awalnya *PHP* adalah singkatan dari "*Personal Home Page Tools*". Selanjutnya diganti menjadi FI (*Forms Interpreter*). Sejak versi 3.0, nama bahasa ini diubah menjadi "PHP: Hypertext Prepocessor" dengan singkatannya "PHP". *PHP* versi terbaru adalah versi ke-5. Berdasarkan survey *Netcraft* pada bulan Desember 1999, lebih dari sejuta *site* menggunakan *PHP*, diantaranya adalah NASA, Mitsubishi, dan *RedHat*. [12]

2.7.5 Framework CodeIgniter

CodeIgniter adalah aplikasi *open source* yang berupa *framework* dengan model *MVC* (*Model, View, Controller*) untuk membangun *website* dinamis dengan menggunakan *PHP*. *CodeIgniter* memudahkan *developer* untuk membuat aplikasi *web* dengan cepat dan mudah dibandingkan dengan membuatnya dari awal. *Codeigniter* adalah sebuah aplikasi *open source* yang bebas untuk digunakan oleh siapapun tanpa harus membayar lisensi untuk menggunakannya. *Codeigniter* juga merupakan sebuah *framework* untuk membangun sebuah aplikasi *website* dinamis menggunakan *PHP* yang dapat digunakan dengan cepat dan mudah tanpa harus membangun apilkasi *PHP* dari awal.

Framework secara sederhana dapat diartikan kumpulan dari fungsi-fungsi/prosedur-prosedur dan *class-class* untuk tujuan tertentu yang sudah siap digunakan sehingga bisa lebih mempermudah dan mempercepat pekerjaan seorang pemrogram, tanpa harus membuat fungsi atau *class* dari awal. Ada beberapa alasan mengapa menggunakan *Framework*:

1. Mempercepat dan mempermudah untuk membangun sebuah *website* atau aplikasi *web*.
2. Proses *maintenance* lebih mudah karena sudah ada skema tertentu dalam sebuah *framework*.

Secara umum *framework* menyediakan fasilitas-fasilitas yang umum dipakai sehingga kita tidak perlu membangun dari awal (misalnya validasi, *pagination*, *multiple database*, *scaffolding*, *session*, *error handling*, dan sebagainya). [12]

2.7.6 Pengertian MVC (*Model View Controller*)

Model View Controller merupakan suatu konsep yang cukup populer dalam pembangunan aplikasi *web*, berawal pada bahasa pemrograman *Small Talk*, *MVC* memisahkan pengembangan aplikasi berdasarkan komponen utama yang membangun sebuah aplikasi seperti manipulasi data, *user interface*, dan bagian yang menjadi kontrol aplikasi. Terdapat 3 jenis komponen yang membangun suatu *MVC pattern* dalam suatu aplikasi yaitu:

1. *View*, merupakan bagian yang menangani *presentation logic*. Pada suatu aplikasi *web*, bagian ini biasanya berupa *file template HTML* yang diatur oleh *controller*. *View* berfungsi untuk menerima dan merepresentasikan data kepada *user*. Bagian ini tidak memiliki akses langsung terhadap bagian model.
2. *Model*, biasanya berhubungan langsung dengan *database* untuk memanipulasi data (*insert*, *update*, *delete*, *search*), menangani validasi dari bagian *controller*, namun tidak dapat berhubungan langsung dengan bagian *view*.
3. *Controller*, merupakan bagian yang mengatur hubungan antara bagian model dan bagian *view*, *controller* berfungsi untuk menerima *request* dan data dari *user* kemudian menentukan apa yang akan diproses oleh aplikasi.

Dengan menggunakan prinsip *MVC* suatu aplikasi dapat dikembangkan sesuai dengan kemampuan *developer*-nya, yaitu *programmer* yang menangani bagian model dan *controller*, sedangkan *designer* yang menangani bagian *view*, sehingga

penggunaan arsitektur *MVC* dapat meningkatkan *maintanability* dan organisasi kode. Walaupun demikian, dibutuhkan komunikasi yang baik antara *programmer* dan *designer* dalam menangani variabel-variabel yang akan ditampilkan. [14]

2.7.7 MySQL

MySQL adalah sebuah perangkat lunak sistem manajemen basis data *SQL* (bahasa Inggris: *database management system*) atau *DBMS* yang *multithread*, *multi-user*, dengan sekitar 6 juta instalasi diseluruh dunia. *MySQL AB* membuat *MySQL* tersedia sebagai perangkat lunak gratis dibawah lisensi *GNU General Public License* (*GPL*), tetapi mereka juga menjual dibawah lisensi komersial untuk kasus-kasus dimana penggunaannya tidak cocok dengan penggunaan *GPL*.

Tidak sama dengan proyek-proyek seperti *Apache*, dimana perangkat lunak dikembangkan oleh komunitas umum, dan hak cipta untuk kode sumber dimiliki oleh penulisnya masing-masing, *MySQL* dimiliki dan disponsori oleh sebuah perusahaan komersial Swedia *MySQL AB*, dimana memegang hak cipta hampir atas semua kode sumbernya. Kedua orang Swedia dan satu orang Finlandia yang mendirikan *MySQL AB* adalah: David Axmark, Allan Larsson, dan Michael "Monty" Widenius. [12]

2.7.8 Aplikasi Android

Android merupakan sistem operasi yang digunakan untuk perangkat mobile berbasis Linux. Pada awalnya sistem operasi ini dikembangkan oleh Android.Inc, yang kemudian dibeli oleh Google pada tahun 2005. Android mengembangkan usaha pada tahun 2007 dibentuklah Open Handset Alliance (OHA), sebuah konsorsium dari beberapa perusahaan, yaitu Texas Instrument, Broadcom Corporation, Google, HTC, Intel, LG, Marvell Technology Group, Motorola, Nvidia, Qualcom, Samsung Electronics, Sprint Nextel, dan T-Mobile dengan tujuan untuk mengembangkan standar terbuka untuk perangkat mobile Smartphone.

Pada tanggal 9 Desember 2008 , ada 14 anggota baru yang akan bergabung di dalam proyek Android, termasuk Packet Video, ARM Holdings, Atheros Communications, Asustek Computer INC, Garmin Ltd, Softbank, Sony Ericsson, Toshiba Corp, dan VodaFone Group Plc (Hermawan, 2010). [3]

2.7.8.1. Jenis-Jenis Android

Android ini juga mempunyai berbagai versi dari awal mula terbentuknya Android sampai sekarang, berikut daftar nama versi android dari awal:[4]

1. Jelly bean v4.1.2

   Android versi Jelly Bean ini dirilis pada 27 Juni 2014 lewat konferensi I/O Google. Jelly Bean menjadi versi Android yang juga banyak mendapatkan update, tercatat 2 kali sudah update dilakukan di Jelly Bean yakni versi 4.1.2 dimana perbedaan dibanding versi sebelumnya adalah segi User Interface yang lebih elegan seta penambahan fitur Google Search.

2. KitKat v4.4.2

   Android Versi KitKat ini paling banyak dipakai pada smartphone masa kini.

3. Lolipop v5.0

   Android Lolipop adalah Android versi terbaru yang diluncurkan Google pada tahun 2014. Versi Lollipop ini pertama kali ditanamkan di Smartphone Google Nexus 6

4. Marsmelow v6.0

   Rilis Terbaru Android 6.0 Bernama Marshmallow | Pasti sobat sudah pada tahu nih OS Smartphone terlaris jaman ini selalu menghadirkan nama nama unik setiap versi yang dirilisnya untuk versi lengkap dari awal sudah pernah saya bahas di Urutan Nama Nama Versi Android Dari Banyak yang mengira bahwa versi android seteleah versi 5.0 Lollipop adalah Milkshake, namun google membantah akan dugaan itu. Google secara resmi mengeluarkan Android versi 6.0 yang diberi nama Marshmallow. Selain itu Android Marshmallow juga akan menambah fitur fitur terbarunya.

5. Nougat v7.0

   Nougat adalah versi Android termutakhir yang baru diperkenalkan pada ajang kumpul developer Google I/O, pertengahan 2016 ini. Beberapa lama setelahnya, Google menghadirkan Nougat secara resmi untuk publik.Pembaruan paling mendasar pada versi Nougat adalah kehadiran Google Assistant yang

menggantikan Google Now. Asisten digital tersebut lebih bisa diandalkan untuk menjalankan berbagai fungsi. Fitur-fitur baru lainnya mencakup layar split-screen saat dipakai multitasking, serta fitur Doze yang telah dikenalkan di versi Android Marshmallow namun telah ditingkatkan. Android Nougat juga memiliki dukungan terhadap platform virtual reality terbaru Google.

6. Oreo v8.0

   untuk pengguna android sekarang lagi booming booming nya nih tentang OS terbarunya, ya itu adalah OS android versi 8.0 atau yang sering juga disebut sebagai Android O atau Android Oreo. Versi android ini resmi diperkenalkan ole Google pada tanggal 22 Agustus 2017 yang lalu dan juga sudah secara resmi bisa diluncurkan langsung ke lapangan, tapi sebelum versi android ini diresmikan oleh google, nama “Oreo” sendiri sudah terendus sejak Android O pertama kali diperkenalkan di ajang Google I/O 2017 pada Mei 2017 lalu.

2.7.9 Android Studio

Android Studio adalah Lingkungan Pengembangan Terpadu - Integrated Development Environment (IDE) untuk pengembangan aplikasi Android, berdasarkan IntelliJ IDEA . 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:

- Sistem versi berbasis Gradle yang fleksibel
- Emulator yang cepat dan kaya fitur
- Lingkungan yang menyatu untuk pengembangan bagi semua perangkat Android
- Instant Run untuk mendorong perubahan ke aplikasi yang berjalan tanpa membuat APK baru
- Template kode dan integrasi GitHub untuk membuat fitur aplikasi yang sama dan mengimpor kode contoh
- Alat pengujian dan kerangka kerja yang ekstensif
- Alat Lint untuk meningkatkan kinerja, kegunaan, kompatibilitas versi, dan masalah-masalah lain

- Dukungan C++ dan NDK
- Dukungan bawaan untuk Google Cloud Platform, mempermudah pengintegrasian Google Cloud Messaging dan App Engine

Setiap proyek di Android Studio berisi satu atau beberapa modul dengan file kode sumber dan file sumber daya. Jenis-jenis modul mencakup:

- Modul aplikasi Android
- Modul Pustaka
- Modul Google App Engine

Secara default, Android Studio akan menampilkan file proyek Anda dalam tampilan proyek Android, Tampilan disusun berdasarkan modul untuk memberikan akses cepat ke file sumber utama proyek Anda.

Semua file versi terlihat di bagian atas di bawah Gradle Scripts dan masing-masing modul aplikasi berisi folder berikut:

- manifests: Berisi file AndroidManifest.xml.
- java: Berisi file kode sumber Java, termasuk kode pengujian JUnit.
- res: Berisi semua sumber daya bukan kode, seperti tata letak XML, string UI, dan gambar bitmap.

Struktur proyek Android pada disk berbeda dari representasi rata ini. Untuk melihat struktur file sebenarnya dari proyek ini, pilih Project dari menu tarik turun Project (dalam gambar 1, struktur ditampilkan sebagai Android).

Anda juga bisa menyesuaikan tampilan file proyek untuk berfokus pada aspek tertentu dari pengembangan aplikasi Anda. Misalnya, memilih tampilan Problems dari tampilan proyek Anda akan menampilkan tautan ke file sumber yang berisi kesalahan pengkodean dan sintaks yang dikenal, misalnya tag penutup elemen XML tidak ada dalam file tata letak.[15]