BAB II
LANDASAN TEORI

2.1 Tinjauan Studi

Marjito, (2016) Aplikasi Penjualan Online Berbasis Android (studi kasus: di Toko Hoax Merch). Terkadang konsumen harus menghubungi terlebih dahulu produk apa yang terbaru tanpa melihat persediaan produk yang di *update* dalam media tersebut. Permasalahan terbesar yang terdapat pada toko Hoax Merch adalah menyinggung efisiensi waktu dalam hal pemesanan. Metode yang digunakan adalah OOSE (*Object Oriented Software Engineering*) dari Ivar Jacobson. Dengan hasil program yang sudah jadi mempermudah dalam penyusunan dan perhitungan pemesanan. Perbedaan dalam penelitian ini adalah aplikasi yang digunakan merupakan aplikasi penjualan online dengan metode OOSE yang serupa dengan penelitian yang akan diajukan dengan aplikasi pemesanan online. [4]

Teguh Surya. (2018) Pengembangan Sistem Aplikasi Percetakan pada PT. Zono Sangangiti Grafika. Masalah yang terjadi yaitu durasi penyelesaian pesanan yang lama. Sebagai contoh, untuk pembuatan 100 buah kalender membutuhkan waktu 1 minggu. Tidak adanya sistem penjadwalan yang tepat menjadi penyebab dari lamanya waktu penyelesaian pesanan. Pesanan dibuat sesuai urutan pemesanan tanpa memperhatikan durasi pengerjaan dari suatu pesanan untuk setiap mesin. Hal ini dapat meningkatkan kemungkinan pesanan selesai menjadi lama. Menggunakan metode algoritma SJF (*shortest job first*) yang dipilih untuk digunakan karena dikira sesuai dengan beberapa metode penjadwalan. Hasil diimplementasikan program sistem yang dikembangkan berbasis web dengan metode SJF bertugas untuk mengatur penjadwalan. [5]

Annisa Febriani. (2017) Penerapan Aplikasi Program Penjualan dan Pembelian Menggunakan Model *Rapid Application Development*. Adanya kesulitan stok pengecekan pakaian yang sudah jadi ditoko, adanya kekeliruan dalam bertransaksi yang dapat menyebabkan kerugian. Lamanya pembuatan laporan yang masih bentuk file excel. Metode yang digunakan metode penelitian *research & development* (R&D) dan metode analisis dan perancangan aplikasinya menggunakan model RAD (*Rapid Application Development*). Hasil yang di capai

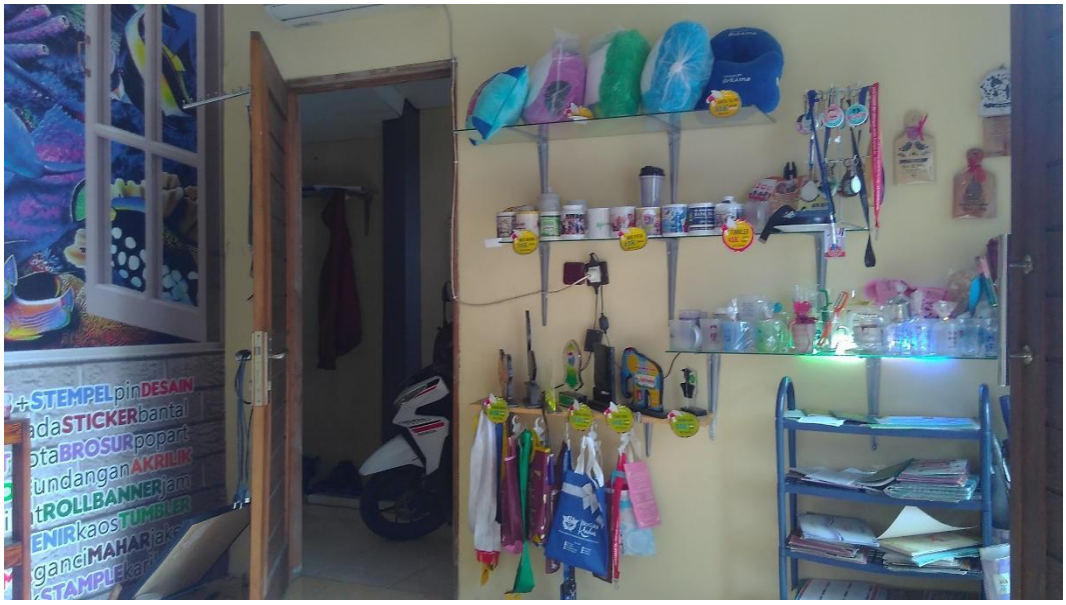
adalah program pembelian dan penjualan pada toko pakaian dengan sistem database yang masih offline. Beberapa toko masih menggunakan program manual dalam transaksinya, seperti membuat laporan kepara pemilik toko. [6]

2.2 Tinjauan Pustaka

2.2.1 Adta Digital

Adta Digital adalah toko yang berdiri sejak 24 November 2015. Adta Digital menyediakan jasa desain, jasa percetakan dan produk yang dipamerkan sangatlah banyak, dikarenakan lokasi yang digunakan kecil dan masih mengontrak. Semenjak berpindah tempat, yang sebelumnya berada di Jalan HOS Cokrominoto Jepara dan sekarang berada di Jalan KS Tubun Demaan Jepara. Lokasi yang baru juga belum dapat menempatkan produk yang di pajang secara keseluruhan. Adta Digital ini buka mulai jam 10.00 Wib sampai jam 17.00 Wib, adta digital tutup pada tanggal merah atau hari besar lainnya.

Gambar 2.1 Toko Adta Digital



Gambar 2.2 Tempat Memamerkan Produk

2.2.2 Android

Dikutip dari buku "Buku Pintar Android" (Masruri, 2015). *Android* merupakan sistem operasi berbasis Linux untuk perangkat *mobile* yang dikutip dari situs Wikipedia. *Android* merupakan sistem operasi gratis dan *open source*, jadi *Android* menyediakan *platform* terbuka bagi para pengembang untuk menciptakan suatu aplikasi sendiri yang mampu berjalan diatas peranti *android*, hal itulah yang menjadikan *android* mampu bersaing di tengah keramaian *smart phone* Blackberry dan Iphone yang lebih dahulu meramaikan pasaran. Namun seiring dengan banyaknya orang-orang canggih di dunia maka semakin canggih pula dunia teknologi. Awalnya *Google Inc* membeli *Android Inc* yang merupakan pendatang baru yang membuat peranti lunak untuk ponsel *smartphone*. 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. Android sebagai platform mobile pertama yang lengkap, terbuka dan bebas maka *android* menjadi pesaing utama dari Apple pada sistem operasi. [7]

Jenis versi Android:

a) Alpha
b) Beta
c) Angel cake: Android Versi 1.0

d) Bettenberg: Android Versi 1.1
e) Cupcake: Android Versi 1.5
f) Donut: Android Versi 1.6
g) Éclair: Android Versi 2.0 - 2.1
h) Froyo: Android Versi 2.2 – 2.23
i) Gingerbread: Android Versi 2.3
j) Honeycomb: Android Versi 3.0 – 3.1
k) Ice Cream Sandwich: Android Versi 4.0
l) Jelly Bean: Android Versi 4.1 – 4.3
m) Kitkat: Android Versi 4.4
n) Lollipop: Android Versi 5.0
o) Marsmallow: Android Versi 6.0
p) Nougat: Android Versi 7.0

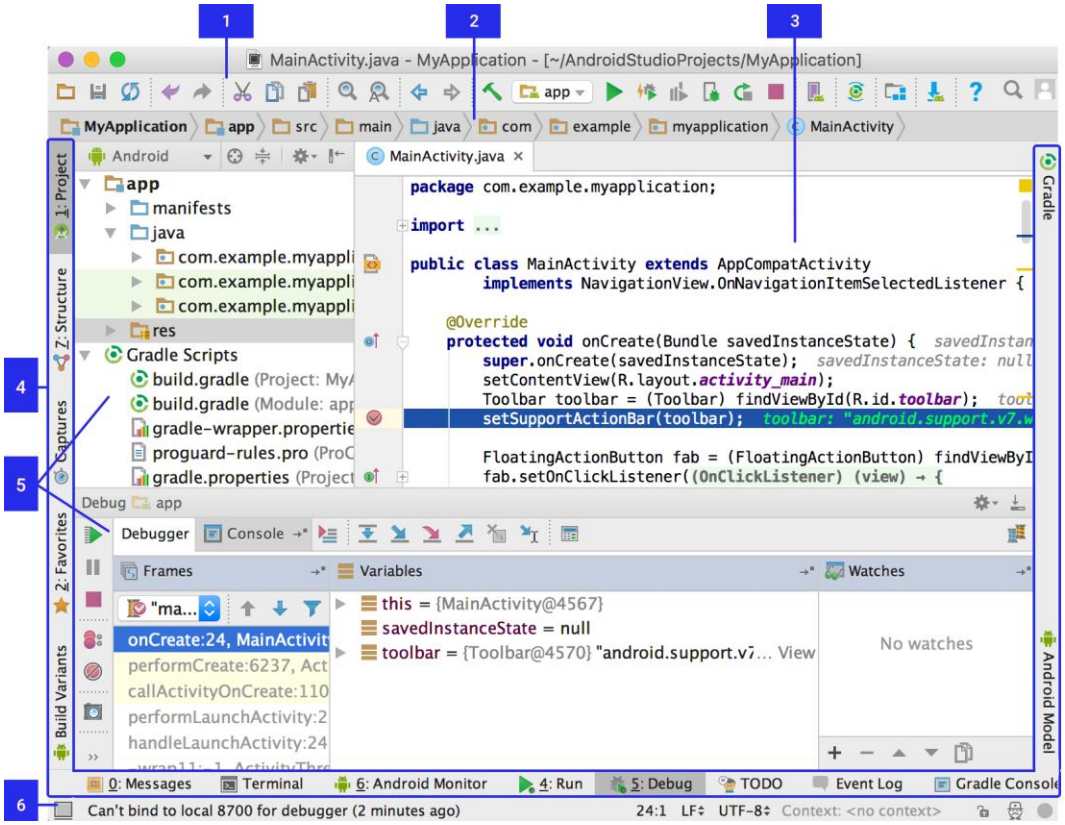
2.2.3 Android Studio

Android Studio adalah sebuah lingkungan pengembangan terpadu untuk mengembangkan pada *platform* Android. *Android Studio* tersedia secara bebas di bawah *Lisensi Apache 2.0.* Berdasarkan *software IDEA JetBrains' IntelliJ, Android Studio* dirancang khusus untuk pengembangan Android. Ini tersedia untuk di-*download* pada *Windows, Mac OS X dan Linux,* dan diganti *Eclipse* Pengembangan *Android Tools (ADT)* sebagai *IDE* utama *Google* untuk pengembangan aplikasi Android asli. Sebelum menggunakan *android studio* diperlukan *java Development kit* yang digunakan untuk *development* saat menulis *code* program. *Android studio* mempunyai *Android Virtual Device (AVD) manager* atau emulator untuk mengelola aplikasi supaya memungkinkan kompatibilitas aplikasi dimasa mendatang. [8]

Jendela utama *android studio* memiliki beberapa bidang yaitu:

1. **Bilah alat** untuk melakukan jenis tindakan, termasuk menjalankan aplikasi.
2. **Bilah navigasi** membantu membuka file untuk diedit dan menampilkan struktur file yang terlihat ringkas.
3. **Jendela editor** tempat membuat dan memodifikasi kode.

4. **Bilah Jendela alat** berisi tombol untuk memungkinkan meluas dan menciutkan jendela alat
5. **Jendela alat** tempat pengelolaan proyek, control versi, penelusuran dan lainnya.
6. **Bilah status** menampilkan proyek status.



Gambar 2.3 Jendela Utama Android Studio

2.2.4 Firebase

Firebase adalah penyedia layanan cloud dengan backend sebagai servis yang berbasis di San Fransisco, California. Perusahaan ini membuat sejumlah produk untuk pengembangan aplikasi mobile ataupun web. Firebase didirikan oleh Andrew Lee dan James Tamplin pada tahun 2011 dan diluncurkan dengan cloud database secara raltime di tahun 2012. Produk utama dari Firebase yakni suatu database yang menyediakan API untuk memungkinkan pengembang menyimpan dan mensinkronisasi data lewat multiple client. Perusahaan ini diakusisi oleh Google pada Oktober 2014. Firebase sendiri sebenarnya

lebih merujuk kepada produk yang mereka namakan dengan nama perusahaan Firebase menyediakan realtime database dan backend sebagai layanan. Suatu aplikasi layanan yang memungkinkan pengembang membuat API untuk disinkronisasikan untuk client yang berbeda-beda dan disimpan pada cloudnya Firebase. Firebase memiliki banyak library yang memungkinkan untuk mengintegrasikan layanan ini dengan Android, iOS, Javascript, Java, Objective-C, dan Node.JS. Firebase memiliki beberapa keunggulan antara lain ada versi gratis, realtime database data terus update setiap milidetiknya, responsive meski offline dan tampilan menarik dan mudah digunakan.

Perbedaan antara Firebase dan MySql antara lain:

1. Firebase bersifat realtime, sedangkan MySql tidak realtime.
2. Dengan firebase tidak perlu membuat kodingan di sisi server.
3. Kodingan di firebase digunakan untuk mengolah data rumit yang rumit kompleks dan lebih banyak.
4. Firebase punya versi berbayar, sedangkan MySql gratis sepenuhnya. [9]

2.2.5 Object Oriented Software Engineering

Object Oriented Software Ongineering (OOSE) adalah suatu rekayasa perangkat lunak yang digunakan untuk membangun sebuah software dengan serangkaian proses terlebuh dahulu. Sedangkan OOSE merupakan salah satu metode dalam perencanaan suatu rekayasa sebelum melakukan perkodean. Metode ini mulai banyak digunakan tetapi konsep Object Oriented ini tidak dapat menjangkau formalitas yang dapat dicapai oleh bahasa spesifikasi formal. konsep ini menggunakan metode UML. Yaitu suatu metode modeling generasi ketiga dan bahasa spesifikasi yang sifatnya *non-proprietary*. Sebenarnya penggunaan dari UML itu sendiri tadak terbatas hanya ada dunia *software modeling*. tetapi bisa pula digunakan untuk *modeling hardware (engineering system)* dan sering digunakan sebagai *modeling* untuk proses bisnis dan juga modeling untuk struktur organisasi. [3]

Kegunaan Object Oriented Software Engineering (OOSE):

1. Sebagai salah satu sumber utama UML. konsep dan notasi dari OOSE telah dimasukkan ke dalam UML.

2. Bagian metodologi OOSE telah berkembang menjadi *Rational Unified Process* (RUP).
3. Alat OOSE telah diganti dengan alat yang mendukung UML dan RUP.
4. OOSE sebagian besar telah digantikan oleh notasi UML dan oleh metodologi RUP.

2.2.6 UML

2.2.6.1 Definisi Unified Modeling Language (UML)

UML merupaka singkatan dari *"Unifed Modelling Language"* adalah Bahasa untuk menspesifikasi, memvisualisasi, membangun dan mendokumentasi sistem perangkat lunak. UML adalah Bahasa pemodelan yang menggunakan konsep orientasi objek atau sebagai suatu Bahasa yang sudah menjadi standar visualisasi perancangan dan dokumentasi sistem software. [10]

UML diaplikasikan untuk maksud tertentu, biasanya antara lain untuk: Merancang perangkat lunak.

a) Sarana komunikasi antara perangkat lunak dengan proses bisnis.
b) Menjabarkan sistem secara rinci untuk analisa dan mencari apa yang diperlukan sistem.
c) Mendokumentasikan sistem yang ada, proses-proses dan organisasinya.

2.2.6.2 Evolusi UML

Menurut Chonoles (Herlawati, 2011:8), Menjelaskan "Bahwa sebelum ada UML, para pengembang bahasa pemograman berorientasi object sulit untuk berkomunikasi satu sama lain." [10]

(Herlawati, 2011:8) Pada bulan oktober 1994, Jim Rumbaugh, penemu notasi Object Modeling Technique (OMT) dan Grady Booch, penemu Booch Method (Metode Booch) bersama-sama menyamakan notasi mereka, dan ditahun yang sama Ivar Jacobson (penemu Objectory Method) ikut bergabung hingga mereka sering disebut "three omigos". [10]

Sejak tahun 1997, divisi Revision Task Force (RTF) milik OMG beberapakali merevisi UML yang dimaksudkan untuk memperkuat konsistensi notasi, meningkatkan kekompakan antara user dan pengembang perangkat lunak.

Akan tetapi UML terpakasa mengikuti perkembangan software-software berbasis objek yang ada (misalnya Java) dari sisi pendekatan komponen (Component-based development) dan kemampuan tools software-software tersebut, setelah dilakukan perubahan secara sistematik, akhirnya dihasilkan UML 2.0 pada tahun 2003.

2.2.6.3 Diagram-Diagram UML

UML menyediakan 9 (sembilan) jenis diagram, yang lain menyebutkan 8 (delapan) karena ada beberapa diagram yang digabung, misalnya diagram komunikasi, diagram urutan dan diagram pewaktuan digabung menjadi diagram interaksi. Namun demikian model-model itu dapat dikelompokkan berdasarkan sifatnya yaitu statis atau dinamis. Dibawah akan disebutkan beberapa jenis diagram dalam UML, diagram tersebut antara lain:

a) Diagram Use-Case, bersifat statis.

Diagram ini memperlihatkan himpunan use-case dan aktor-aktor (suatu jenis khusus darikelas). Diagram ini terutama sangat penting untuk mengorganisasi dan meodelkan perilaku suatu sistem yang dibutuhkan seperti diharapkan pengguna.

b) Diagram Interaksi dan Sequence (urutan), bersifat dinamis.

Diagram urutan adalah diagram interaksi yang menekankan pada pengiriman pesan dalam suatu waktu tertentu.

c) Diagram Aktivitas (activity Diagram), bersifat dinamis.

Diagram aktivitas adalah tipe khusus dari diagram status yang memperlihatkan aliran tipe khusus dari diagram status yang memperlihatkan aliran dari suatu aktivitas ke aktivitas lainnya dalam suatu sistem. Diagram ini terutama penting dalam permodelan aliran kendali antar objek.

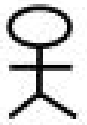
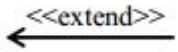
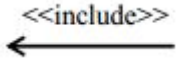
d) Diagram Kelas, bersifat statis

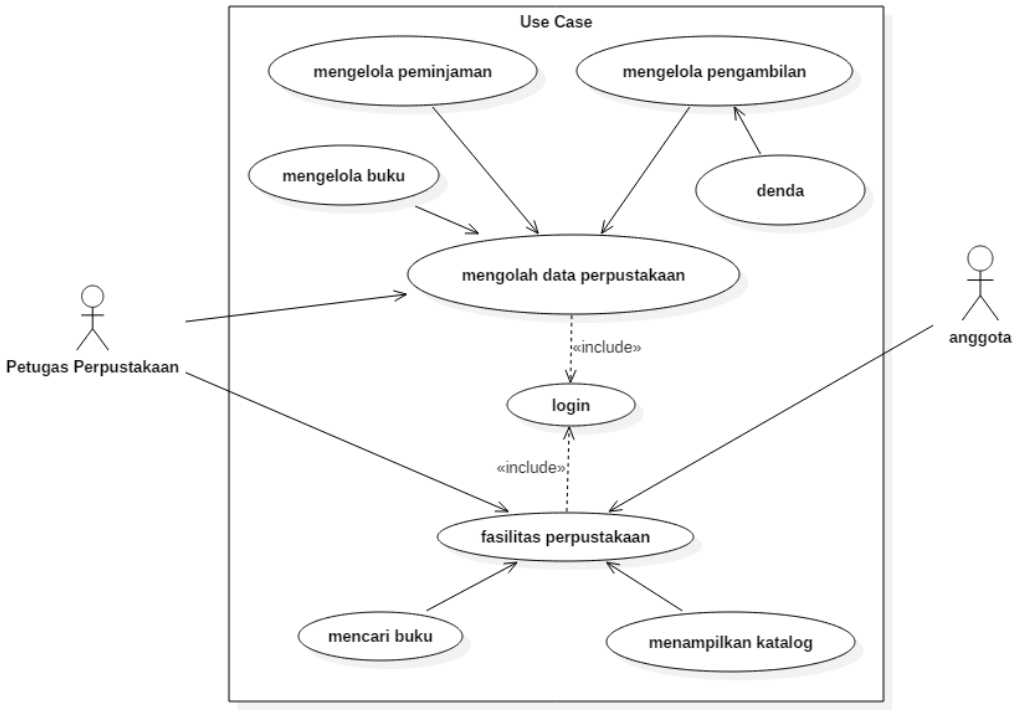
Diagram ini memperlihatkan himpuan kelas-kelas, atar muka-antar muka, kolaborasi-kolaborasi, serta relasi-relasi. Diagram ini umum dijumpai pada pemodelan sistem berorientasi objek, meskipun bersifat statis, sering pula diagram kelas memuat kelas-kelas aktif. [10]

2.2.7 Use Case Diagram

Use case diagram dibawah ini menjelaskan graphical dari beberapa atau semua actor, use case dan interaksi diantara komponen-komponen tersebut yang memperkenalkan suatu sistem yang akan dibangun. [10]

Tabel 2.1 Usecase Diagram

No	Gambar	Nama	Keterangan
1		Aktor	Mewakili peran orang, sistem yang lain, alat ketika berkomunikasi dengan *use case*
2		Use case	Interaksi antara sistem dan actor
3		Association	Penghubung antara actor dengan *use case*
4		Extend	Untuk menunjukkan *use case* seluruhnya merupakan fungsional dari *use case* lainnya
5		Include	Untuk menunjukkan *use case* merupakan tambahan fungsional dari *use case* lainnya jika suatu kondisi terpenuhi
6		Generalisasi	Untuk menunjukkan spesialisasi actor untuk dapat berpartisipasi dengan *use case*.



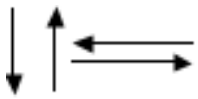
Gambar 2.4 Contoh Usecase Diagram

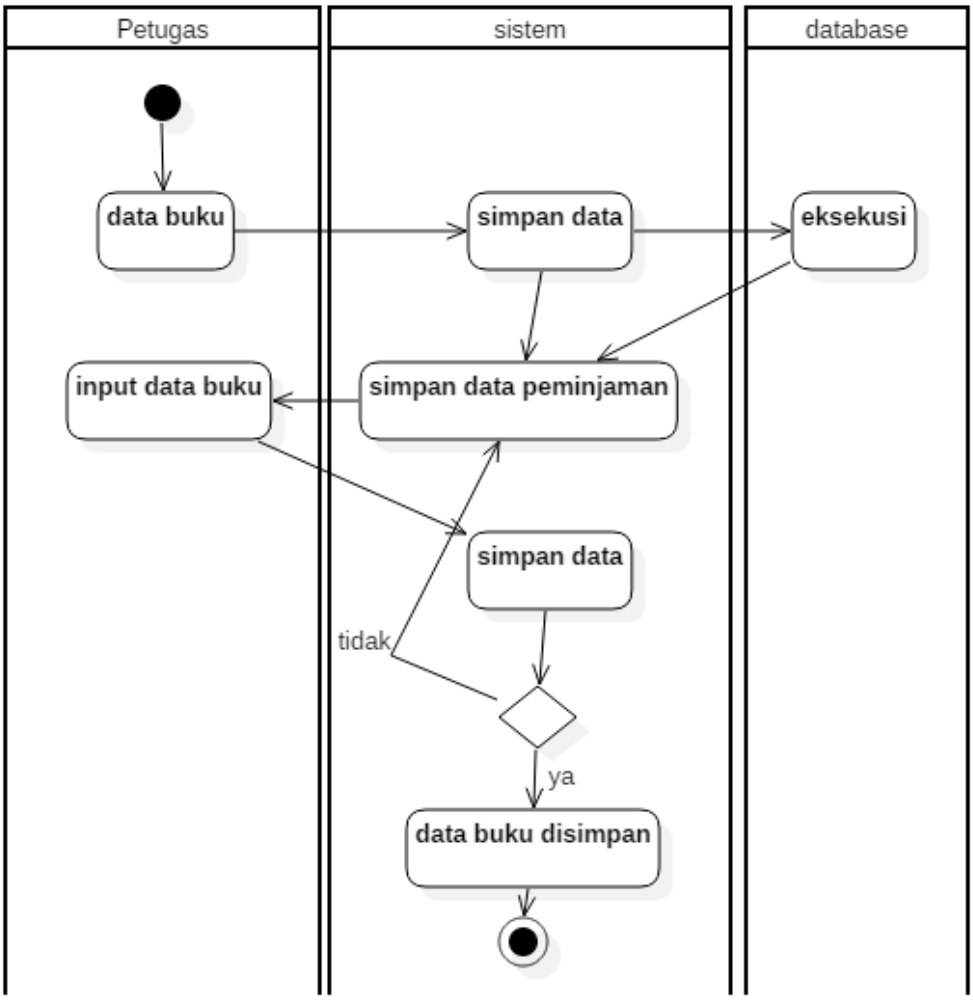
2.2.8 Activity Diagram

Dari gambar Class Diagram dibawah ini dapat dijelaskan bahwa activity diagram menggambarkan aliran kerja atau aktivitas dari sebuah sistem. [10]

Tabel 2.2 Activity Diagram

No	Gambar	Nama	Keterangan
1		*Initial Node*	Titik awal
2		*Activity Final Node*	Titik akhir
3		*Activity*	Memperlihatkan bagaimana masing-masing kelas antarmuka saling berinteraksi satu sama lain

4		*Action*	State dari sistem yang mencerminkan eksekusi dari suatu aksi
5		*Decision*	Digunakan untuk menggambarkan suatu keputusan/tindakan
6		*Line Connector*	Digunakan untuk menghubungkan satu symbol dengan symbol lainnya

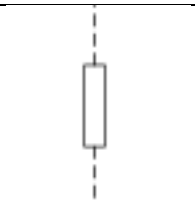
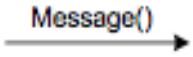


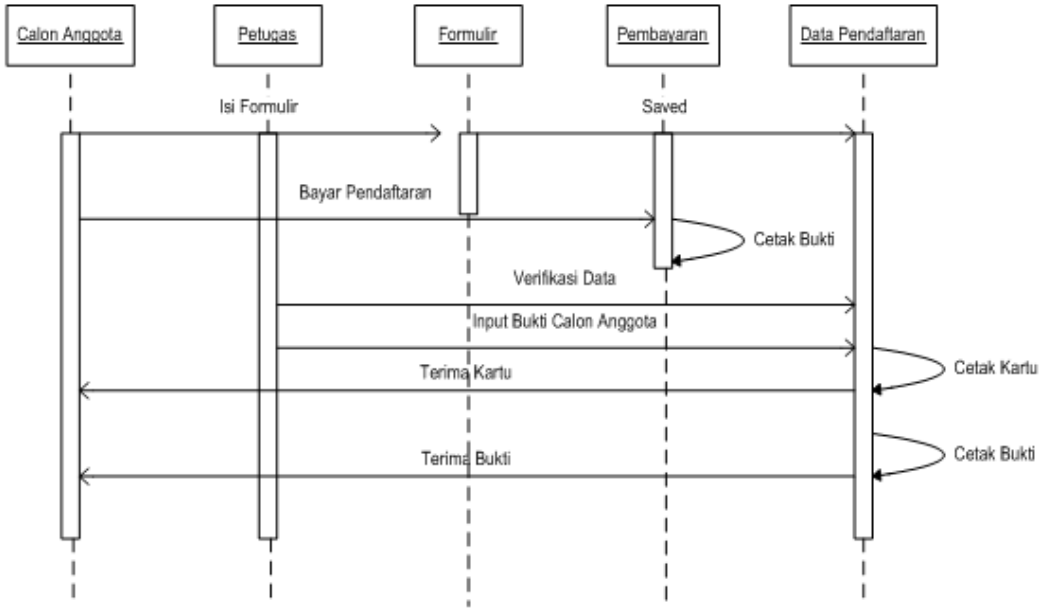
Gambar 2.5 Contoh Activity Diagram

2.2.9 Sequence Diagram

Diagram sequence menggambarkan kelakukan objek pada use case dengan mendeskripsikan waktu hidup objek dan message yang dikirimkan dan diterima anatar objek. Oleh karena itu untuk menggambarkan diagram sequnce maka harus diketahui objek-objek yang terlibat dalam sebuah use case beserta metode-metode yang dimiliki kelas yang diinstansiasi menjadi objek itu. [11]

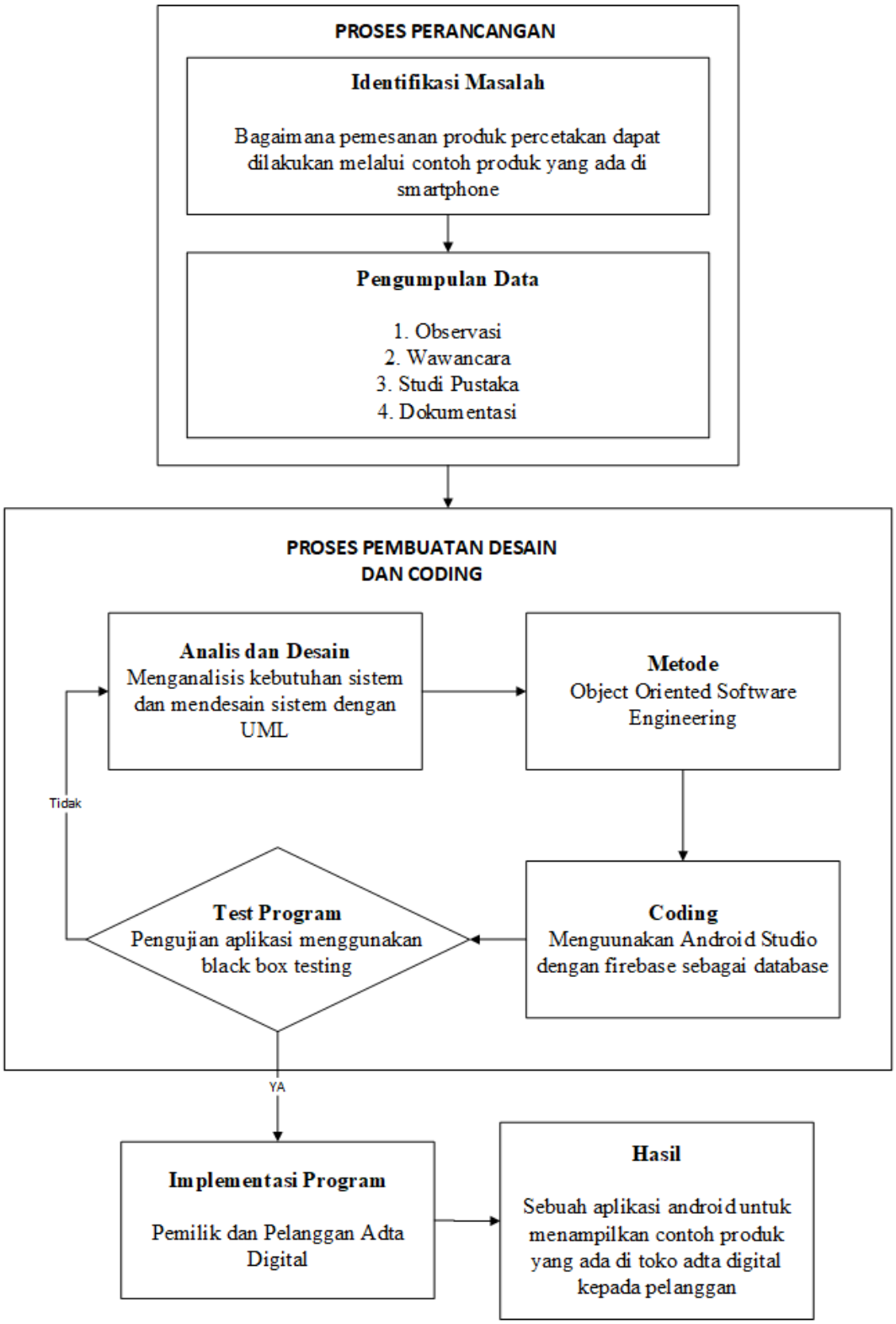
Tabel 2.3 Sequence Diagram

No	Gambar	Nama	Keterangan
1		*LifeLine*	Objek entity, antarmuka yang saling berinteraksi
2		Actor	Digunakan untuk menggambarkan user/pengguna.
3		*Message*	Spesifikasi dari komunikasi antar objek yang memuat informasi tentang aktifitas yang terjadi.
4		*Boundary*	Digunakan untuk menggambarkan sebuah form.
5		*Control Class*	Digunakan untuk menghubungkan boundary dengan tabel
6		*Entity Class*	Digunakan untuk menggambarkan hubungan kegiatan yang akan dilakukan



Gambar 2.6 Contoh Sequence Diagram

2.3 Kerangka Pemikiran



Gambar 2.7 Kerangka Pemikiran