

BAB II

LANDASAN TEORI

2.1 Studi Kepustakaan

Studi kepustakaan ini memberi tambahan ilmu bagi penulis dalam melakukan penelitian, oleh karena itu penulis mengambil referensi dari jurnal ilmiah / skripsi sebagai acuan pembuatan tugas akhir skripsi ini, jurnal / skripsi yang menjadi referensi penulis memiliki kasus yang berbeda-beda.

Berdasarkan penelitian yang dilakukan oleh Rahmad Juari (2016) membahas tentang aplikasi yang dapat melakukan multi target (menampilkan 2 object sekaligus) dengan adanya aplikasi ini maka peminat *design racing stripes* atau *sticker* lebih tertarik karena hasil desain dapat ditampilkan dalam bentuk 3D. Pada aplikasi tersebut masih menggunakan *markerless* atau *marker barcode* untuk menampilkan suatu object 3D dan hasil *design* yang di rancang dalam aplikasi tersebut menampilkan design beberapa mobil.

Penelitian yang dilakukan oleh Remo Prabowo dkk (2015) dengan aplikasi ini, pengguna dapat mengetahui keragaman rumah tradisional Indonesia melalui kamera smartphone android dengan menggunakan KTP sebagai marker yang dapat memudahkan pengguna dalam mengakses aplikasi. Aplikasi yang dibuat berhasil memodernisasi media pengenalan rumah adat Indonesia dan meningkatkan antusiasme masyarakat untuk mengenal rumah adat Indonesia, adanya Interaksi menggunakan tombol yang disediakan pada Aplikasi ini memudahkan *User* untuk menjalankan aplikasi.

Penelitian berikutnya yang dilakukan oleh Juan Nicky Aristo Pattymahu dan Oktoverano Lengkong (2016) penelitian ini membuat tentang aplikasi virtual eksplorasi RSUP Prof Dr. R. D. Kandou Manado diharapkan dapat memudahkan pengunjung dalam mencari ruangan atau lokasi yang ada di lingkungan rumah sakit. Peneliti akan membuat aplikasi virtual eksplorasi ini menggunakan Google Sketchup sebagai tool untuk pemodelan objek 3D bangunan rumah sakit dan Unity 3D yang akan digunakan sebagai game engine. Penelitian ini akan menghasilkan aplikasi virtual eksplorasi di RSUP Prof Dr. R. D. Kandou Manado menggunakan game engine berbasis Android.

Berdasarkan studi kepustakaan diatas orisinalitas pada penelitian ini yaitu pada penelitian pertama terletak pada menampilkan object 3d. Penelitian ke dua dan ketiga terletak pada permasalahan yang mau diteliti. Maka dilakukan penelitian lebih lanjut dengan judul “Aplikasi *Sticker* Motor Scoopy Berbasis Android (Studi Kasus CV. Upgrade Graphic Design)”.

2.2 Konsep Teori

2.2.1 Sticker

Secara umum *sticker* adalah lembaran kecil kertas atau plastik yang ditempelkan atau *sticker* juga sering dikatakan suatu bahan yang dapat menempel dengan sendiri atau memiliki perekat. *Sticker* terdiri dari dua lapis, lapisan pertama merupakan bidang untuk membuat gambar dan lapis kedua yaitu kertas yang berfungsi untuk melindungi bagian perekatnya. Secara *visual*, *sticker* terdiri dari dua yaitu *sticker transparent* dan *sticker non-transparent*.

Dengan adanya sticker sebagai bahan dasar untuk pembuatan *sticker*, sangat banyak pengguna otomotif yang menginginkan motor mereka didesain sedemikian rupa agar terlihat lebih menarik. Bahkan sesuai dengan perkembangan teknologi informasi dan promosi, sticker semakin banyak digunakan sebagai media untuk promosi, seperti branding atau sekedar untuk memeperindah suatu objek.

2.2.2 Skuter Matik

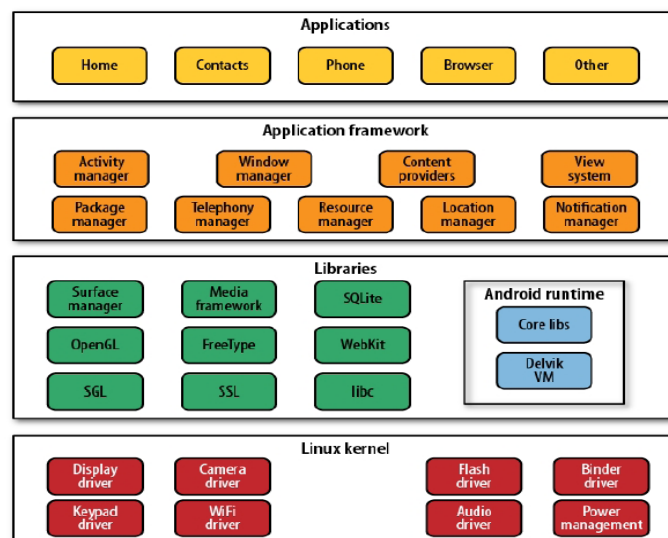
Tipe ini adalah tipe sepeda motor otomatis yang tidak menggunakan operan gigi manual dan hanya cukup dengan satu akselerasi, sepeda motor ini memiliki kapasitas silinder (CC) kecil dan posisi pengemudi yang tegak, ukuran sepeda motor ini lebih kecil dan ringan daripada tipe bebek. Sepeda motor ini memiliki ruang kosong di antara kemudi dan pengendara yang memungkinkan untuk kaki bisa diletakan di tempat tersebut. Sepeda motor ini sangat cocok untuk wanita dan ini digunakan untuk keperluan dalam kota/wilayah. Sepeda motor tipe ini memiliki dimensi ukuran ban dan roda yang cukup kecil. Contoh sepeda motor tipe ini yaitu: Honda Beat, Honda Vario, Honda Scoopy, Honda Spacy Helm-in, Vespa Piaggio, Yamaha Mio, dll.

2.2.3 Android

Android merupakan sistem operasi yang berbasis *Linux*. *Android* dikembangkan oleh Google Inc pada saat Google membeli pertama kali pada tahun 2005. Google secara resmi pada tahun 2007 Open *Handset Allience* mengumumkan bahwa *Android* telah menjadi *open source*, sehingga semua orang

dapat mengembangkan aplikasi untuk *Android*. Setelah itu, pada tahun 2008 *Android SDK 1.0* dirilis untuk pertama kalinya.

Software Development Kit (SDK) merupakan perangkat lunak yang diperlukan untuk membuat aplikasi Android dengan menggunakan bahasa pemrograman Java. Walaupun menggunakan bahasa pemrograman Java, Android tidak menggunakan *Java Virtual Machine* (JVM) seperti aplikasi Java pada umumnya. *Android* mempunyai *Virtual Machine* sendiri yang disebut *Dalvik Virtual Machine* yang merupakan software stack. Dalvik adalah *virtual machine* dengan tujuan pembuatan khusus untuk Android, yang dikembangkan oleh Dan Bornstein dan timnya di Google. Adapun gambar kerangka konseptual aplikasi seperti gambar 2.1 dibawah ini.



Gambar 2.1 Kerangka Konseptual Aplikasi

Pada Gambar 2.1 menjelaskan tentang gambaran dari struktur Android dan dapat dilihat bahwa Android dibagi menjadi 5 bagian utama, yaitu Applications,

Application framework, Libraries, Android runtime dan Linux kernel. Bagian-bagian tersebut akan dijelaskan lebih lanjut.

1. Application

Pada bagian ini, berisi aplikasi utama yang dimiliki Android. Pertama adalah Home, Contacts, Phone, Browser, dan Others. Semua aplikasi ditulis dalam bahasa Java, sehingga pengembang dapat memodifikasi sebanyak yang diinginkan.

2. Application framework

Pengembang memiliki akses penuh terhadap Application Programming Interface(API) yang digunakan dalam aplikasi ini. API sendiri adalah sebuah interface yang diimplementasikan untuk interaksi dengan program lain.

3. Libraries

Android memiliki satu set libraries inti yang menjalankan aplikasi, tujuannya agar pengembang dapat mengaksesnya secara langsung.

4. Android Runtime

Virtual Machine dan Library Core. Sebagian besar fungsinya tersedia di Java Library Core dan disediakan Android Runtime.

5. Linux Kernel

Linux Kernel ini berisi keypad, WiFi, kamera, dan driver Kernel Linux yang melindungi semua struktur internal Android dalam satu paket, seperti motor pendukung antara hardware dan software.

2.2.4 Aplikasi Game Engine Unity 3D 5.6.2f1

Unity Engine suatu game engine yang terus berkembang. Engine ini merupakan salah satu game engine dengan lisensi source proprietary, namun untuk lisensi pengembangan dibagi menjadi 2, yaitu free (gratis) dan berbayar sesuai perangkat target pengembangan aplikasi (Berta Sihite, Febriliyan Samopa, 2013). Game engine merupakan komponen yang ada dibalik layar setiap video game. Adapun fitur-fitur yang dimiliki oleh Unity 3D antara lain sebagai berikut.

- a) Integrated development environment (IDE) atau lingkungan pengembangan terpadu.
- b) Penyebaran hasil aplikasi pada banyak platform.
- c) Engine grafis menggunakan Direct3D (Windows), OpenGL (Mac, Windows), OpenGL ES (iOS), and proprietary API (Wii).
- d) Game Scripting melalui Mono. Scripting yang dibangun pada Mono, implementasi open source dari NET Framework. Selain itu Pemrogram dapat menggunakan UnityScript (bahasa kustom dengan sintaks JavaScript-inspired), bahasa C# atau Boo (yang memiliki sintaks Python-inspired).

Unity mendukung pengembangan aplikasi Android. Sebelum dapat menjalankan aplikasi yang dibuat dengan Unity Android diperlukan adanya pengaturan lingkungan pengembang Android pada perangkat. Untuk itu pengembang perlu men-download dan menginstal SDK Android dan menambahkan perangkat fisik ke sistem.

2.2.5 Tiga Dimensi (3D)

3D adalah sebuah objek yang memiliki panjang, lebar, dan tinggi yang memiliki bentuk. 3D tidak hanya digunakan dalam matematika dan fisika saja melainkan pada bidang grafis, seni, animasi, komputer dan lainlain. 3D dapat menggambarkan setiap objek yang terjadi pada tiga sumbu sistem koordinat cartesian. Sebuah sistem koordinat Cartesian pada dasarnya adalah cara mudah menggambarkan sumbu X dan Y. Dalam dunia 2D terdapat dua sumbu, X untuk sumbu horisontal dan Y untuk sumbu vertikal, hal yang sama juga terdapat dalam dunia 3D, yaitu memiliki sumbu untuk koordinat yang akan digambar, tetapi dengan satu pengecualian, ada sumbu ketiga yaitu sumbu Z, yang mewakili kedalaman.



(a) 2D



(b) 3D

Gambar 2.2 a. Gambar 2D b. Gambar 3D

Pada Gambar 2.2 bisa dilihat perbedaan antara gambar 2D (kiri) dan gambar 3D (kanan), pada gambar 2D dapat dilihat gambar tersebut hanya terdiri dari dua sumbu, yaitu sumbu X untuk lebar, dan sumbu Y untuk tinggi. Sedangkan pada gambar 3D juga memiliki ruang seperti lemari. Istilah “3D” juga digunakan untuk menunjukkan representasi dalam grafika komputer (digital),

penggunaan 3D dalam grafika komputer dapat dipandukan dengan gambar 2D sebagai tekstur dari objek 3D yang dibuat.

2.2.6 Aplikasi Blender 3D

Menurut Lance Flavell (2010) Blender merupakan paket aplikasi pemodelan dan animasi tiga dimensi yang memiliki berbagai fungsi yang tidak dimiliki aplikasi tiga dimensi lainnya. Blender juga semacam program yang dapat melakukan berbagai fungsi.

- a. Blender adalah aplikasi pemodelan tiga dimensi yang dapat membuat sebuah karakter untuk film.
- b. Blender memiliki sebuah alat yang kuat untuk pewarnaan permukaan model.
- c. Blender memiliki sebuah fasilitas dalam rigging dan animasi yang sangat kuat. Model tiga dimensi yang dibuat dapat dirancang untuk bergerak dan beraksi sedemikian rupa.
- d. Blender memiliki mesin rendering sendiri dan dapat dianggap layaknya studio pencahayaan yang lengkap untuk sebuah film.
- e. Tidak seperti paket aplikasi 3D lainnya, Blender memiliki compositing module sendiri, sehingga hasil live shoot bisa langsung di masukkan dan diintegrasikan dengan model tiga dimensi. Blender juga memiliki editor pengurutan video yang unik, sehingga memungkinkan untuk memotong dan mengedit video tanpa harus bergantung pada aplikasi pihak ketiga tambahan untuk tahap editing akhir produksi.

- f. Selain semua itu, Blender juga memiliki fasilitas Game Engine.

2.2.7 C# (C Sharp)

C# (dibaca: C sharp) merupakan sebuah bahasa pemrograman yang berorientasi objek yang dikembangkan oleh Microsoft sebagai bagian dari inisiatif kerangka .NET Framework. Bahasa pemrograman ini dibuat berbasiskan bahasa C++ yang telah dipengaruhi oleh aspek - aspek atau pun fitur bahasa yang terdapat pada bahasa - bahasa pemrograman lainnya seperti Java, Delphi, Visual Basic dan lain - lain dengan beberapa penyederhanaan. Menurut standar ECMA-334 C# Language Specification, nama C# terdiri atas sebuah huruf latin C (U+0043) yang diikuti oleh tanda pagar yang menandakan angka # (U+0023). Tanda pagar # yang digunakan memang bukan tanda kres dalam seni musik (U+266F), dan tanda pagar # (U+0023) tersebut digunakan karena karakter kres dalam seni musik tidak terdapat didalam keyboard standar. (Jonathan, 1998).

2.2.8 Unified Modeling Language (UML)

Secara umum Unified Modeling Language (UML) merupakan “bahasa” untuk visualisasi, spesifikasi, konstruksi, serta dokumentasi. Dalam kerangka visualisasi, para pengembang menggunakan UML sebagai suatu cara untuk mengkomunikasikan idenya kepada para pemrogram serta calon pengguna sistem/perangkat lunak. Dengan adanya “bahasa” yang bersifat standar, komunikasi perancang dengan pemrogram (lebih tepat lagi komunikasi antar

anggota kelompok pengembang) serta calon pengguna diharapkan menjadi mulus (Nugroho, 2005).

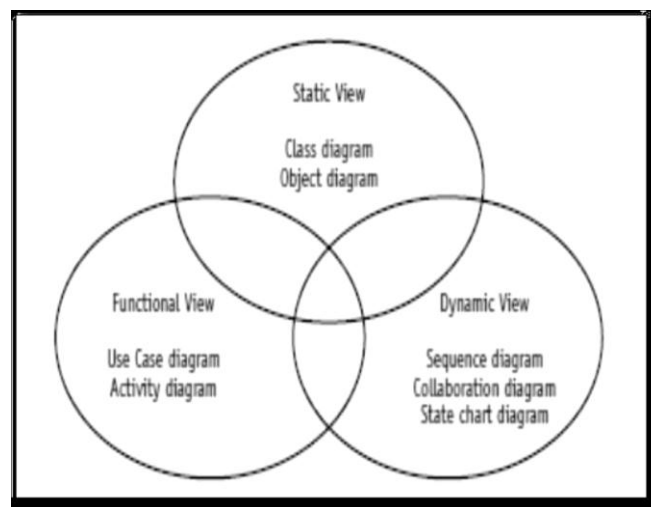
Dalam kerangka spesifikasi, UML menyediakan model-model yang tepat, tidak ambigu, serta lengkap. Secara khusus, UML menspesifikasikan langkah-langkah penting dalam pengambilan keputusan analisis, perancangan, serta implementasi dalam sistem yang sangat bernuansa perangkat lunak (software intensive system). Dalam hal ini, UML bukanlah merupakan bahasa pemrograman tetapi model-model yang tercipta berhubungan langsung dengan berbagai macam bahasa pemrograman, sehingga adalah mungkin melakukan pemetaan (mapping) langsung dari model-model yang dibuat dengan UML ke bahasa-bahasa pemrograman berorientasi objek.

Secara umum UML diterapkan dalam pengembangan sistem/perangkat lunak berorientasi objek sebab metodologi UML ini umumnya memiliki keunggulan-keunggulan dibawah ini (Nugroho, 2005):

1. *Uniformity*. Dengan metodologi UML (atau metodologi berorientasi objek pada umumnya), para pengembang cukup menggunakan 1 metodologi dari tahap analisis hingga perancangan. Hal ini tidak bisa dilakukan dalam metodologi pengembangan terstruktur. Dengan perkembangan masa kini ke arah aplikasi GUI (Graphical User Interface), UML juga memungkinkan kita merancang komponen antarmuka pengguna (User Interface) secara terintegrasi bersama dengan perancangan perangkat lunak sekaligus dengan perancangan basis data.

2. *Understandability*. Dengan metodologi ini kode yang dihasilkan dapat diorganisasi kedalam kelas-kelas yang berhubungan dengan masalah sesungguhnya sehingga lebih mudah dipahami siapa pun juga.
3. *Stability*. Kode program yang dihasilkan relatif stabil sepanjang waktu sebab sangat mendekati permasalahan sesungguhnya di lapangan.
4. *Reusability*. Dengan metodologi berorientasi objek, dimungkinkan penggunaan ulang kode, sehingga pada gilirannya akan sangat mempercepat waktu pengembangan perangkat lunak (sistem informasi).

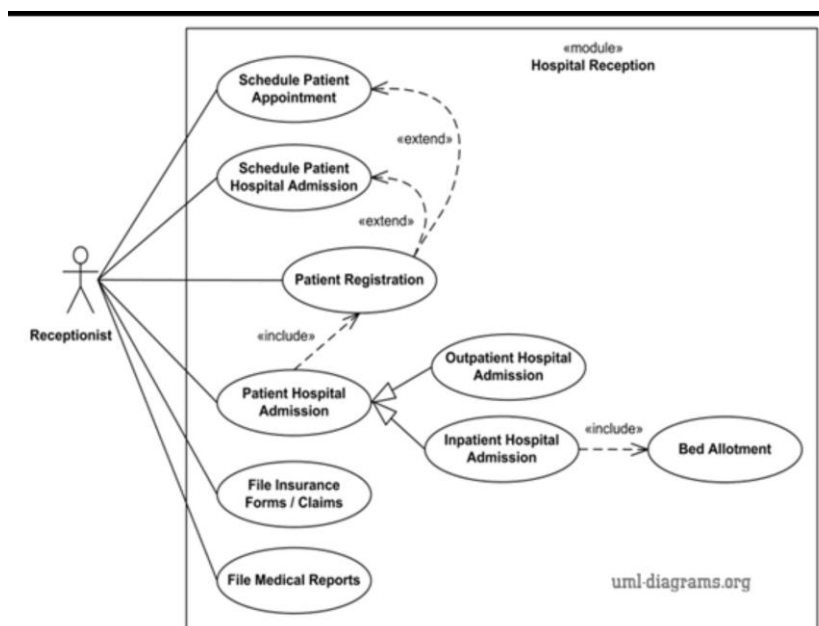
UML terdiri atas pengelompokan diagram-diagram sistem. Diagram adalah yang menggambarkan permasalahan maupun solusi dari permasalahan suatu model. Salah satu cara untuk mengatur diagram UML adalah dengan menggunakan *view*. *View* adalah kumpulan dari diagram yang menggambarkan aspek yang sama dari proyek yang terdiri dari *Static View*, *Dinamis View*, dan *Fungsional View* (Pender, 2002). Gambar 2.3 menggambarkan sifat komplementer dari tiga pandangan dan diagram yang membuat setiap tampilan.



Gambar 2.3 Three complementary views or sets of diagrams (Pender, 2002)

1. Usecase Diagram

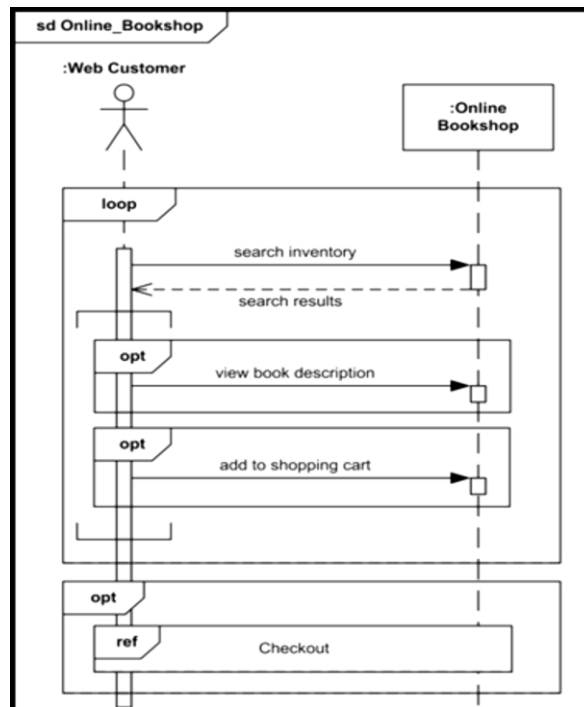
Bentuk dari diagram usecase dapat terlihat pada gambar 2.2 dibawah ini. Bersifat statis, diagram ini memperlihatkan himpunan usecase dan aktor-aktor (suatu jenis khusus dari kelas). Diagram ini terutama sangat penting untuk mengorganisasi dan memodelkan perilaku dari suatu sistem yang dibutuhkan serta diharapkan pengguna.



Gambar 2.4 Usecase Diagram

2. Sequence Diagram

Bersifat dinamis, bentuk dari diagram sequence dapat terlihat pada gambar 2.3 dibawah ini. Diagram urutan adalah diagram interaksi yang menekankan pada pengiriman pesan (message) dalam suatu waktu tertentu.



Gambar 2.5 Sequence Diagram

2.2.9 Pengujian Sistem

Pengujian pada dasarnya adalah menemukan serta menghilangkan ‘bug’ (kesalahan-kesalahan) yang ada di sistem/perangkat lunak. Kesalahan-kesalahan itu dapat diakibatkan beberapa hal utama, antara lain kesalahan saat penentuan spesifikasi sistem, kesalahan saat melakukan analisis permasalahan, kesalahan saat perancangan, serta kesalahan saat implementasi.

2.2.9.1 Teknik Pengujian Sistem

Konsep kualitas sangat penting demi kepuasan pengguna (juga pengembang). Untuk mencapai kualitas yang diharapkan dari sistem yang kita kembangkan pada umumnya ada beberapa strategi pengujian yang dapat dilakukan. Strategi-strategi itu adalah (Bahrawi, 1999 dalam Nugroho, 2005):

1. *Black-Box Testing*. Pada pengujian ini kita tidak perlu tahu apa sesungguhnya terjadi pada sistem/perangkat lunak. Yang kita uji adalah masukkan serta keluarannya. Artinya, dengan berbagai masukkan yang kita berikan, apakah sistem memberikan keluaran seperti yang kita harapkan?
2. *White-Box Testing*. Pengujian jenis ini mengasumsikan bahwa spesifikasi logika adalah penting dan perlu dilakukan pengujian untuk menjamin apakah sistem berfungsi dengan baik. Tujuan utama dari strategi pengujian ini adalah pengujian berbasis kesalahan.
3. *Top-Down Testing*. Pengujian ini berasumsi bahwa logika utama atau interaksi antar objek perlu diuji lebih lanjut. Strategi ini seringkali dapat mendeteksi cacat/kesalahan/kekurangan yang serius. Pendekatan ini sesuai dengan strategi pengujian berbasis scenario.

2.2.9.2 Pengukuran Tingkat Kepuasan Pengguna

Mengenai kepuasan pengguna kita masih harus meninjau seberapa jauh sistem yang kita kembangkan memuaskan pengguna. Beberapa cara yang dapat ditempuh untuk mengetahui kepuasan pengguna adalah:

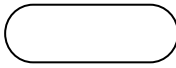

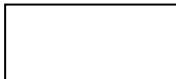
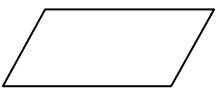
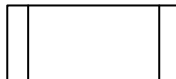
1. Wawancara. Wawancara diperlukan untuk mengetahui tingkat kepuasan pengguna dengan mengajukan beberapa pertanyaan.
2. Kuesioner. Kuesioner merupakan daftar pertanyaan yang diajukan pada seorang responden untuk mencari jawaban dari permasalahan yang diteliti.

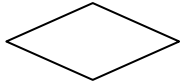


3. Pengamatan Langsung. Pengamatan langsung bisa juga dilakukan untuk mengetahui apakah sistem yang kita kembangkan sesuai dengan kebutuhan serta harapan pengguna.

2.2.10 Diagram Alir (*Flowchart*)

Flowchart (diagram alir) dapat digunakan sebagai alternatif untuk menyajikan algoritma. Algoritma adalah sekumpulan langkah yang rinci yang ditunjukkan untuk menyelesaikan suatu masalah. Flowchart ini menunjukkan setiap langkah program atau prosedur dalam urutan yang tepat saat terjadi. Flowchart adalah bentuk penyajian grafis yang menggambarkan solusi langkah demi langkah terhadap suatu permasalahan (Abdul Kadir, 2013). Simbol standar untuk flowchart (diagram alir) dapat dilihat pada Tabel 2.1.

Tabel 2.1 Simbol *Flowchart*

No	SIMBOL	NAMA	FUNGSI
1		<i>Terminator</i>	Permulaan/akhir program
2		Garis Alir <i>(Flow Line)</i>	Arah aliran program
3		Proses	Proses perhitungan/proses pengolahan data
4		<i>Input/Output Data</i>	Proses input/output data, parameter, informasi
5		<i>Predefined Process (Sub Program)</i>	Permulaan sub program/proses menjalankan sub program

7		<i>Decision (Keputusan)</i>	Perbandingan pernyataan, penyeleksian data yang memberikan pilihan untuk langkah selanjutnya
8		<i>On Page Connector</i>	Penghubung bagian-bagian flowchart yang berada pada satu halaman
9		<i>Off Page Connector</i>	Penghubung bagian-bagian flowchart yang berada pada halaman berbeda