BAB II
LANDASAN TEORI

2.1 *Visualisasi*

Visualisasi adalah rekayasa dalam pembuatan gambar diagram atau animasi untuk penampilan suatu informasi. Secara umum visualisasi dalam bentuk gambar baik yang bersifat abstrak maupun nyata, Pada saat ini visualisasi telah berkembang dan banyak dipakai untuk keperluan ilmu pengetahuan rekayasa visualisasi disain produk, pendidikan, multimedia, interaktif dan kedokteran Perkembangan bidang animasi juga telah membantu banyak dalam bidang visualisasi yang lebih kompleks dan canggih.

2.2 *Latte Art*

Latte art adalah seni melukis di atas kopi. Kopi *lovers* saat kita ngopi-ngopi di *caffe* dan kita memedan kopi *latte* , kita tidak akan mendapati secangkir kopi latte polos tanpa hiasan. Hiasan khas yang memperindah kopi kita ini disebut *latte art*. Waktu Peracik kopi atau barista sedang beraksi membuatnya di *coffee bar*, kita pasti takjub melihatnya, Saat para barista ini memeragakan melukis kopi dengan susu yang dibuat sedemikian rupa dengan lukisan atau gambar- gambar yang menarik kelihatan sangat mudah, tetapi itu tak semudah yang di bayangkan mereka perlu ketrampilan khusus dan science agar latte art terlihat sempurna.

2.2.1 Teknik membuat Kopi Latte art

Ada beberapa teknik yang digunakan saat melukis kopi *latte art*, namun yang paling sering digunakan dengan dua cara Teknik *free pour* dan Teknik *etching* teknik ini lah yang paling sering digunakan para barista untuk melukis di atas kopi teknik inipun sering di temui di *caffe shop*

1. Teknik free pour

   Teknik yang pertama untuk membuat *latte art* adalah *free pour*, yaitu teknik yang hanya bertujuan memanipulasi jatuhnya *latte* ke dalam kopi. cara membuatnya adalah dengan menuang *latte* dari sisi cangkir yang lama-kelamaan menuju ujung cangkir lain. Cara ini paling sederhana karena tidak membutuhkan alat untuk melukis. Hasilnya biasanya bisa di berbentuk di antaranya bunga atau bentuk hati. Brikut gambar 2.2.1 Teknik free pour



(Gambar 2.1 Teknik free pour)

2. Teknik etching

   Teknik ini cenderung lebih sulit dibanding teknik free pour, karena teknik etching menuntut peracik kopi atau barista untuk menunjukkan kreativitas seni yang dimiliki di atas busa dengan campuran sirup atau bubuk cokelat. Jika menggunakan teknik ini, maka diperlukan alat bantu berbentuk stik agar bisa membuat pola yang pas dan menarik. Berikut gambar 2.2 Teknik etching

(Gambar 2.2 Teknik etching)

2.3 Android

Menurut Safaat Android adalah sistem operasi berbasis Linux bagi telepon seluler seperti telepon pintar dan komputer tablet. Android juga menyediakan *platform* terbuka bagi para pengembang untuk menciptakan aplikasi mereka sendiri yang akan digunakan untuk berbagai macam piranti gerak. Awalnya, Google Inc. membeli Android Inc., pendatang baru yang membuat piranti lunak untuk ponsel. kemudian dalam pengembangan Android, dibentuklah Open Handset Alliance, konsorsium dari 34 perusahaan piranti keras, piranti lunak, dan telekomunikasi, termasuk Google, HTC, Intel, Motorola, Qualcomm, T-Mobile, dan Nvidia.

2.3.1 Versi Android

1. Android versi 4.0 (ICS: Ice Cream Sandwich)

   Pada tanggal 19 Oktober 2011, diperkenalkannya Android versi 4.0 yang membawa fitur Honeycomb untuk *smartphone*, menambahkan fitur baru termasuk membuka kunci dengan pengenalan wajah, jaringan data pemantauan penggunaan dan kontrol, kontak jaringan sosial terpadu, perangkat tambahan fotografi, pencarian email secara *offline*, dan berbagi informasi dengan menggunakan NFC. Ponsel pertama yang menggunakan sistem operasi ini adalah Samsung Galaxy Nexus.

2. Android versi 4.1 (Jelly Bean)

   Android Jelly Bean yaang diluncurkan pada acara Google I/O membawa sejumlah keunggulan dan fitur baru. Adapun penambahan fitur baru diantaranya yaitu meningkatkan input keyboard, desain baru fitur pencarian,

UI yang baru dan pencarian melalui *Voice Search* yang lebih cepat. Google Now yang juga menjadi bagian yang diperbarui pun tak ketinggalan. Google Now memberikan informasi yang tepat pada waktu yang tepat pula. Salah satu kemampuannya adalah dapat memberikan sebuah informasi tentang teknik pembuatan *latte art* Sistem operasi Android Jelly Bean 4.1 muncul pertama kali dalam produk tablet Asus, yakni Google Nexus 7.

2.3.2 Fitur Android

Adapun beberapa fitur-fitur menurut Safaat (2012) yang tersedia di
Android adalah sebagai berikut ini :

1. Kerangka aplikasi: memungkinkan penggunaan dan penghapusan komponen yang tersedia.
2. Dalvik mesin virtual: mesin virtual dioptimalkan untuk perangkat telepon seluler.
3. Grafik: grafik di 2D dan grafis 3D berdasarkan pustaka OpenGL.
4. SQLite: untuk penyimpanan data.
5. Mendukung media: audio, video, dan berbagai format gambar (MPEG4, H.264, MP3, AAC, AMR, JPG, PNG, GIF).
6. GSM, Bluetooth, EDGE, 3G, 4G dan WiFi (tergantung piranti keras).
7. Kamera, *Global Positioning System* (GPS), kompas, NFC dan *accelerometer* (tergantung piranti keras).

2.4 Adobe Flash Profesional CS6

Andi (2013) mendefinisikan bahwa *Adobe Flash Professional CS6* merupakan perangkat lunak multiguna yang dapat dimanfaatkan untuk berbagai macam animasi dengan berbagai fitur canggih yang ada didalamnya dan juga dapat menggambar, membuat animasi, hingga digunakan untuk membuat *game* yang menarik dan berkualitas. Area kerja *Adobe Flash Professional CS6* dirancang secara khusus agar ruang kerja yang digunakan dapat diatur dan lebih mudah dipahami oleh pemakai pemula maupun para desainer *flash* yang telah berpengalaman.

Adobe Flash Professional CS6 adalah program yang cukup kompleks sehingga tidak mungkin untuk menjabarkannya secara lengkap. Disini hanya akan diberikan sedikit pengertian dan fungsi dasar dari *Adobe Flash Professional CS6.*

2.5 Adobe Integrate Runtime

Wahana Komputer (2014) Adobe AIR adalah sebuah cross operating system runtime yang dikembangkan oleh Adobe sehingga memungkinkan pengembang memanfaatkan keterampilan mereka (seperti Flash, Flex, HTML, Javascript, dan PDF) untuk membangun RIA (*Rich Internet Application*) dan kontennya ke dalam platform baru.

2.6 Action Script 3.0

Madcoms (2012), mendefinisikan bahwa a*ctionscript* yaitu bahasa pemrograman pada *adobe flash player* dan *adobe AIR*. Dimana bahasa pemrograman ini bisa interaktif, menangani data, dan banyak digunakan pada *Flash*, *Flex*, dan *AIR* baik konten maupun aplikasi. *Actionscript* dapat di eksekusi oleh *Actionscript Virtual Machine* (*AVM*), yang mana ini merupakan bagian dari *Flash* dan *AIR*. Bahasa pemrograman ini akan familiar dengan para *develover* dengan kemampuan yang minim terhadap OOP (*Object Oriented Programming*).

Menurut Madcoms (2012), a*ctionscript* 3.0 mempunyai kapabilitas yang lebih bagus dari pada *Actionscript* sebelumnya. Yaitu di desain untuk memfasilitasi kreasi atau pekerjaan yang mempunyai kompleksitas yang tinggi dalam membangun aplikasi dengan data set yang besar dan *ObjectOriented*, *code bases* yang bisa dipakai berulang kali. Bahasa pemrograman ini dapat mengeksekusi 10 kali lebih cepat dari pada *actionscript* sebelumnya.

2.7 Adobe Photoshop CS6

Henky Prihatna (2005), mendefinisikan bahwa *photoshop* adalah salah satu *software* pengolah *grafik* yang banyak digunakan oleh para *desainer grafis* dan *web* di seluruh dunia. Tampilanya yang mudah difahami, kelengkapan fasilitas yang ditawarkan, serta kemudahan memperoleh fasilitas pendukung dari berbagai sumber menjadikan *Photoshop* menjadi pilihan paling handal bagi para desainer.

2.8 Metode Pengembangan Perangkat Lunak

Pressman (2013) metode prototype sebagai suatu paradigma dimulai dengan perancangan, analisis, desain, implementasi dan penggunaan selanjutnya mengidentifikasi Metode adalah ilmu yang digunakan untuk memperoleh kebenaran menggunakan penelusuran dengan tata cara tertentu dalam menemukan kebenaran, tergantung dari realitas yang sedang dikaji. Metode juga bisa disebut sebagai suatu teknik/cara untuk mengerjakan sebuah pekerjaan dengan urutan-urutan tertentu atau prosedur-prosedur tertentu. Metode prototype dibagi dalam 5 sistem antara lain :

1. Perancangan

   Perancangan sistem adalah proses multilangkah yang fokus pada desain pembuatan program perangkat lunak termasuk struktur data, arsitektur perangkat lunak, representasi antar muka, dan prosedur pengkodean. Tahap ini mentranslasi kebutuhan perangkat lunak dari tahap analisis kebutuhan ke representasi desain agar dapat diimplementasikan menjadi program pada tahap selanjutnya

2. Analisis

   Analisis dan definisi persyaratan adalah proses pengumpulan kebutuhan dilakukan secara intensif untuk mespesifikasikan kebutuhan perangkat lunak agar dapat dipahami perangkat lunak seperti apa yang dibutuhkan oleh user. Spesifikasi kebutuhan perangkat lunak pada tahapini perlu untuk didokumentasikan.

3. desain

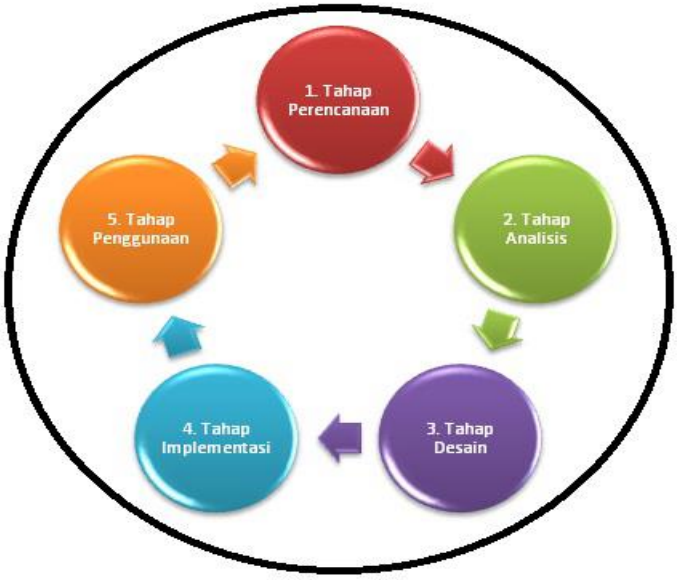
   Perancangan sistem dan perangkat lunak ditranslasikan ke dalam program perangkat lunak. Hasil dari tahap ini adalah program komputer sesuai dengan desain yang telah dibuat pada tahap desain.

4. Implementasi

   Pengujian fokus pada perangkat lunak secara dari segi lojik dan fungsional dan memastikan bahwa semua bagian sudah diuji. Hal ini dilakukan untuk meminimalisir kesalahan *(error)* dan memastikan keluaran yang dihasilkan sesuai dengan yang diinginkan.

5. penggunaan

   Tidak menutup kemungkinan sebuah perangkat lunak mengalami perubahan ketika sudah dikirimkan ke *user*. Perubahan bisa terjadi karna adanya kesalahan yang muncul dan tidak terdeteksi saat pengujian atau perangkat lunak harus beradaptasi dengan lingkungan baru. Tahap pendukung atau pemeliharaan dapat mengulangi proses pengembangan mulai dari analisis spesifikasi untuk perubahan perangkat lunak yang sudah ada, tapi tidak untuk membuat perangkat lunak baru. Sistem model *prototype* dapat dilihat pada gambar 2.3



Gambar 2.3 Sistem Model *prototype*

Kelebihan dari metode *prototyping* ini sebagai berikut :

1. Adanya komunikasi yang baik antara pengembang dan pelanggan
2. Pengembangan dapat bekerja lebih baik dalam menentukan kebutuhan pelanggan.
3. Lebih menghemat waktu dalam pengembangan sistem.
4. Penerapan menjadi lebih mudah karna pemakaian mengetahui apa yang diharapkannya.

Kekurangan dari metode prototype ini sebagai berikut :

1. Resiko tinggi yaitu untuk masalah-masalah yang tidak terstruktur dengan baik, ada perubahan yang besar dari waktu ke waktu, dan adanya persyaratan data yang tidak menentu.
2. Interaksi pemakai penting. Sistem harus menyediakan dialog online antara pelanggan dan komputer.
3. Hubungan pelanggan dengan komputer yang disediakan mungkin tidak menceritakan teknik perancangan yang baik.

2.8.1 Unified Modeling Language (UML)

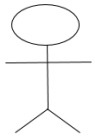
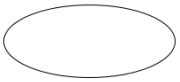
Yasin (2012) mendefinisikan *Unified Modelling Language* (UML) adalah sebuah "bahasa" yang telah menjadi standar dalam industri untuk visualisasi, merancang dan mendokumentasikan sistem piranti lunak, UML menawarkan sebauh standar untuk merancang model sebuah sistem. Tujuan Pengunaan UML yaitu untuk memodelkan suatu sistem yang menggunakan konsep berorientasi objek dan menciptakan bahasa pemodelan yang dapat digunakan baik oleh manusia maupun mesin.

Sedangkan menurut Whitten & Bentley (2007) UML adalah suatu konvensi pemodelan yang digunakan untuik menspesifikasikan atau mendeskripsikan sebuah sistem piranti lunak yang terkait dengan objek. UML terdiri dari beberapa tipe diagram, yaitu:

a. *Use-Case* Diagram

Menurut Whitten & Bentley (2007), *use-case* diagram adalah diagram yang menggambarkan interaksi antara sistem dan pengguna. Dengan kata lain, diagram ini mendeskripsikan siapa yang akan menggunakan sistem itu dengan cara apa pengguna berinteraksi dengan sistem. Simbol Simbol Use Case Diagram dapat dilihat pada Table 2.1

Tabel 2.1 Simbol Use Case Diagram

Simbol	Keterangan
	Aktor : Seseorang atau sesuatu yang berinteraksi dengan sistem yang sedang dikembangkan.
	*Use case* : perungkat tertinggi dari fungsionalitas yang dimiliki sistem.
	*Association :*adalah relasi antara actor dan *use case.*
	Generalisasi : untuk memperlihatkan struktur pewaris yang terjadi.

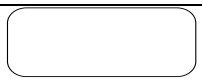
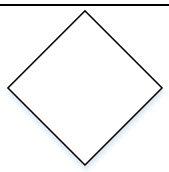
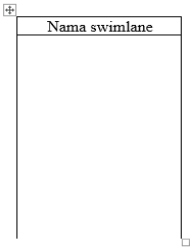
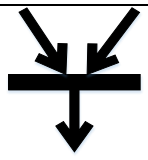
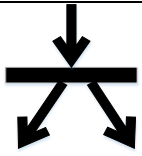
b. Activity Diagram

Menurut Whitten & Bentley (2007), *Activity diagram* adalah sebuah diagram yang dapat digunakan untuk mendeskripsikan secara grafis aliran kerja dari sebuah proses bisnis, langkah-langkah dari sebuah *use case*, atau logika dari sebuah perilaku objek (*object behavior*/*method*). *Activity diagram* adalah representasi grafis dari aliran kerja (*workflow*) atas aktivitas-aktivitas yang bertahap dan aksi-aksi di dalam sebuah sistem.

Diagram ini juga mendukung proses seleksi, pengulangan (iterasi), dan konkurensi. *Activity diagram* sejenis dengan *state diagram*/*statechart diagram*. Diagram ini mendeskripsikan/menggambarkan kumpulan aktivitas dengan menunjukkan sekuensial dari aktivitas yang dilakukan. Diagram ini

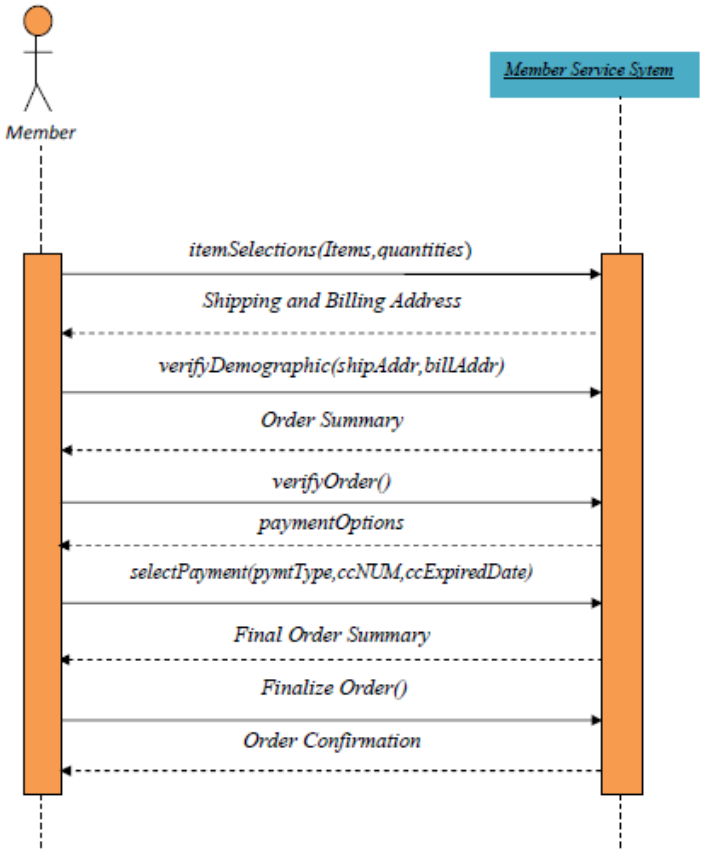
juga bisa menunjukkan aktivitas yang bersifat kondisional atau paralel. Berikut notasi yang digunakan pada *activity diagram*: Simbol Simbol *Activity Diagram* dapat dilihat pada Tabel 2.2

Tabel 2.2 Simbol *Activity Diagram*

Simbol	Keterangan
	*Activity* : Memperlihatkan bagaimana masing-masing kelas antarmuka saling berinteraksi satu sama lain.
	*Initial Node* : Bagaimana objek dibentuk atau diawali
	*Actifity Final Node* : Bagaimana objek dibentuk dan diakhiri.
	*Decision* : Asosiasi percabangan dimana jika ada pilihan aktifitas lebih dari satu.
	*Swimlane* : Memisalkan organisasi bisnis yang bertanggung jawab terhadap aktifitas yang terjadi.
	*Join* : Digunakan untuk menunjukkan kegiatan yang digabungkan.
	*Fork* : Digunakan untuk menunjukkan kegiatan yang dilakukan secara paralel

c. *Sequence Diagram*

Sequence Diagram menggambarkan bagaimana obyek berinteraksi satu sama lain melalui pesan di dalam pelaksanaan suatu *use case* atau operasi (Whitten & Bentley 2007). Dunia berbasis obyek berjalan dengan saling mengirim pesan diantara obyek. Sistem dari *sequence Initial Node* Bulatan merah muda di dalam lingkaran menggambarkan awal dari proses. *Actions* Bulatan persegi panjang menggambarkan langkah individu. *Flow* Panah di dalam *Diagram* menunjukan alur proses. *Decision* Bentuk berlian dengan 1 *flow* yang masuk dan 2 atau lebih *flow* yang keluar. *Flow* yang keluar ditandai untuk mengindikasikan kondisinya. *Activity final* bulatan merah muda dengan garis hitam berbentuk lingkaran menandakan akhir dari proses. *diagram* membantu untuk memulai mengindetifikasi *high-level messages.* Contoh sistem Sequence Diagram dapat dilihat pada Gambar 2.8



Gambar 2.8 *Contoh Sequence Diagram*

Actor		Yang memulai yang diambil dari *use case* digambarkan dengan *use case* aktor simbol.
System		kotak mengindikasikan sistem sebagai "*black box*". Merupakan standar notasi pada *sequence Diagram* yang mengindikasikan *instance* yang berjalan dari sistem.
Lifelines		garis *vertical* putus-putus ke arah bawah dari *actor* dan simbol, mengindikasikan hidup di *sequence*.
Activation bars		persegi panjang yang berada pada *lifelines* mengindikasikan periode waktu peserta aktif dalam interaksi.
Input messages		panah horizontal dari aktor ke sistem mengindikasikan pesan masuk.
Output messages		panah horizontal dari sistem ke aktor digambarkan dengan garis putus-putus.

Gambar 2.9 Keterangan *Sequence Diagram*