BAB 2
TINJAUAN PUSTAKA

2.1 Ruang Lingkup Objek Penelitian

Objek penelitian yang sedang diteliti adalah Sekolah Menengah Atas, yaitu bertempat di SMA Negeri 1 Bandung.

2.1.1 Sejarah SMA Negeri 1 Bandung

Pada tahun 1950 berdiri sebuah SMA yang dikenal dengan sebutan SMA PARKI (Pasundan) di bawah pimpinan Bapak Among Praja. Tahun pelajaran 1950/1951 atau tepatnya tanggal 1 agustus 1950, SMA PAKRI menjadi SMA Negeri. Sejak itu SMA PAKRI menjadi SMA NEGERI 3 A/B Bandung dengan dua jurusan, yaitu bagian A dan bagian B yang dipimpin oleh ibu Sutjinah. Pada tahun 1964 SMA Negeri 1 Bandung masih dikenal dengan nama SMA 1A. Saat itu SMA ini di pimpin oleh Drs. Abdullah Djumantradja. Pada tahun 1953 Kepemerintahan Pendidikan dan Kebudayaan memutuskan bahwa SMA III dibagi dua, yaitu : MA IIIA Bagian Bahasa dipimpin oleh Pak MI. Kartadipradja dan SMA IIIB Bagian Ilmu Pasti dipimpin oleh Pak Tjetje Djajadisastra.

Pada tahun 1956, sekolah mendapat pergantian nama SMA Negeri IIIA menjadi SMA Negeri IA, sedangkan SMA Negeri IIIB menjadi SMA Negeri IVB. SMA IA dipindahkan ke daerah Bandung Utara, menempati sebuah gedung Lyceum di Jl. Dago. Sedangkan SMA IVB berubah status menjadi SMA Negeri IV. Pada tahun 1965 untuk pertama kalinya SMAN 1A membuka jurusan Ilmu Hayat, Ilmu Sosial dan Sastra Budaya. Sesuai dengan penamban jurusan yang didasarkan pada peraturan pemerintah, nama SMAN IA pun diubah menjadi SMA Negeri 1 Bandung.

SMA Negeri 1 Bandung yang terletak di Jalan Ir. H. Juanda No. 93 Bandung, memiliki areal 8.450 m2 dan luas bangunannya 3.790 m2, sampai tahun 2011/2012 memiliki sarana dan prasarana sebagai berikut : 29 ruang belajar, Ruang Kepala Sekolah, Ruang Wakasek, Ruang Staf Wakasek, Ruang Guru, Ruang Tata Laksana, Ruang Pengolahan Data, Ruang Perpustakaan, Ruang BK, Ruang Laboratorium, Ruang Multimedia, Ruang Aula dan Masjid.

Adapun prestasi akademik yang pernah diraih oleh SMA Negeri 1 Bandung adalah Juara I tingkat Internasional APEC Future Education Festival di Gyeongju Korea Selatan pada tahun 2012, Juara II Olimpiade Fisika tingkat Nasional pada tahun 2011, Juara I ASEAN Science Enterprise Challenge pada tahun 2010, Juara Tkt. Provinsi Olimpiade Kebumian, Juara 1 Honda DBL pada tahun 2010, Juara II Masjid Mushola Award pada tahun 2010.

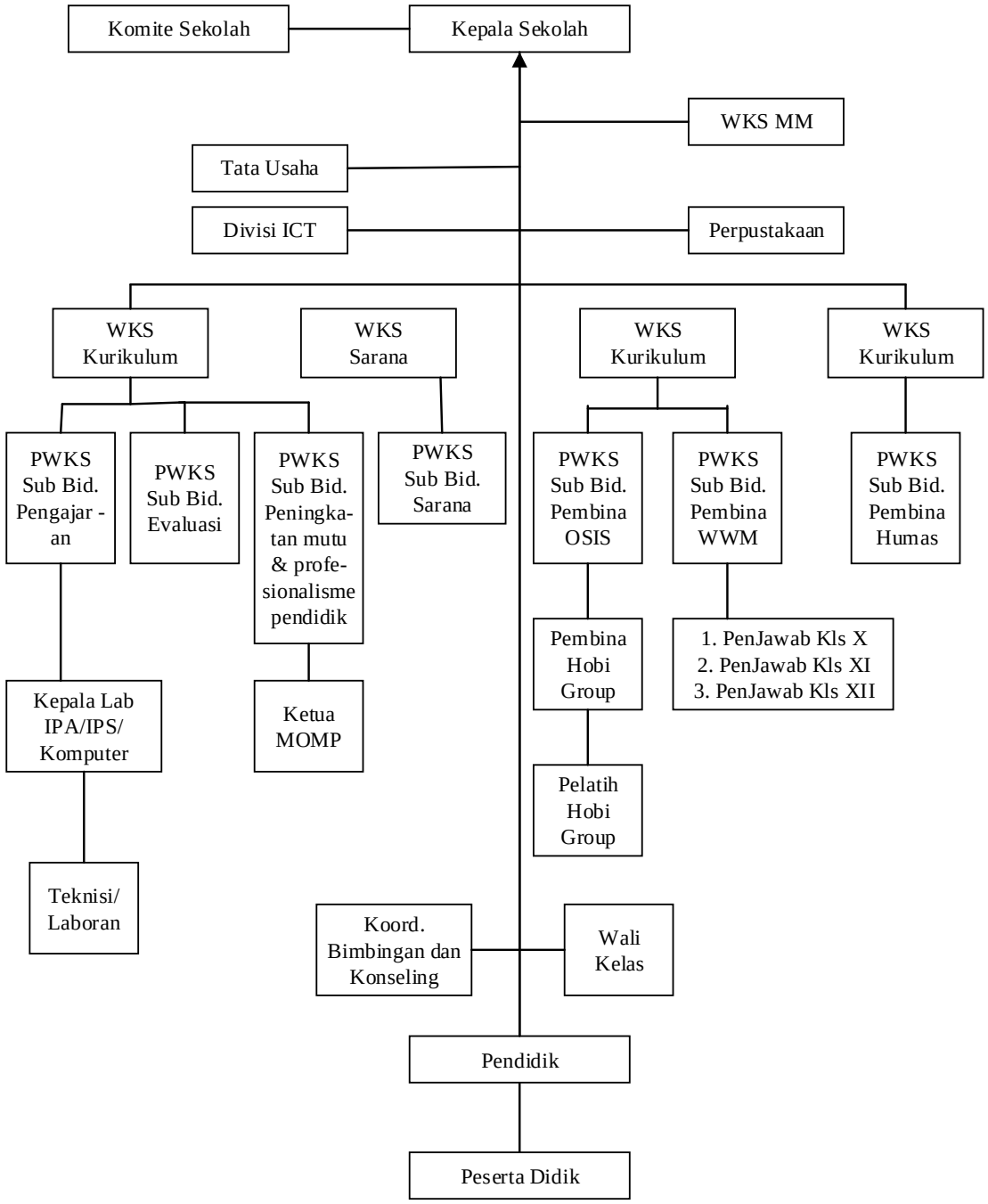
2.1.2 Visi dan Misi

Visi yang diemban oleh SMA Negeri 1 Bandung adalah "**Mewujudkan Sumber Daya Manusia yang Berprestasi dan Berbudi Pekerti Baik Sesuai dengan Tujuan Pendidikan Nasional**". Adapun misi SMA Negeri 1 Bandung :

a. Meningkatkan ketakwaan kepada Tuhan Yang Maha Esa.
b. Meningkatkan rasa kebangsaan dan nasionalisme.
c. Mengembangkan seni budaya lokal dan nasional.
d. Meningkatkan pengetahuan dan perilaku yang berwawasan lingkungan.
e. Melaksanakan pendidikan dan pembelajaran, bimbingan dan konseling, Serta layanan administrasi secara efektif dan efesien, profesional, akuntabel dan berdaya saing yang berbasis *Information and Communication Technology* (ICT).

2.1.3 Struktur Organisasi SMA Negeri 1 Bandung

Struktur Organisasi pada SMA Negeri 1 Bandung dapat dilihat pada gambar 2.1.



Gambar 2.1 Struktur Organisasi

2.2 Landasan Teori

Landasan teori merupakan sebuah penjabaran dari teori-teori pendukung, karena di dalam pengembangan suatu perangkat lunak memerlukan teori-teori pendukung untuk menjabarkan suatu sistem dibuat. Teori yang mendukung dalam pembangunan sistem ini merupakan sebuah teori tentang pembuatan game, perancangan system, rpg maker vxace, photoshop, paintool sai dan microsoft vsio.

2.2.1 Definisi *Game*

Pergertian *game* adalah aktivitas yang dilakukan untuk *fun* atau menyenangkan yang memiliki aturan sehingga ada yang menang dan ada yang kalah [7]. Selain itu, *game* membawa arti sebuah kontes, fisik atau mental, menurut aturan tertentu, untuk hiburan, rekreasi, atau untuk menang taruhan.

2.2.2 Pengertian *Game Engine*

Engine game adalah komponen software inti dari sebuah video *game*.n*Engine game* biasanya menangani perenderan grafik dan teknologi penting lainnya,tetapi juga memungkinkan dalam menangani fungsi tambahan seperti kecerdasan buatan,pendeteksian tubrukan antara objek *game* dengan benda lainnya [7]. Bedasarkan Tujuan penggunaannya, klasifikasi game dibagi menjadi tiga bagian, yaitu :

1. *Game as Game*

   Game yang dimaksud adalah game untuk kesenangan atau *fun*.
2. *Game as Media*

   Tujuan utama dari *game as media* adalah untuk menyampaikan pesan tertentu,menyampaikan pesan dari pembuat game tersebut. Contoh umum dari *game as media* adalah game yang termasuk dalam serious game, seperti *advergame*, *edutainment* atau *edugame*, exergame, simulasi, dan lainnya.
3. *Game Beyond Game*

   Bisa disebut juga dengan istilah *gamification. Gamification* adalah penerapan konsep atau cara berpikir game design ke dalam lingkup non-game. Bentuk gamification sendiri pun biasanya tidak berupa game,

misalnya sistem reward dari poin yang dikumpulkan dengan berbelanja, atau lainnya.

2.2.3 Jenis-Jenis *Game*

Berdasarkan jenisnya, *game* terdiri dari beberapa kategori [7], yaitu :

a. Side Scrolling

Game tipe ini dapat dikatakan tipe-tipe game klasik, game yang hanya memiliki dua arah pergerakan antara kanan dan kiri.

b. Racing

Game dengan tipe simulasi balapan baik itu dalam bentuk kendaraan bermotor atau kendaraan dalam dunia fantasi.

c. Fighting

Game bertipe *fighting* ini merupakan salah satu tipe game yang cukup digandrungi dikalangan remaja, dalam game ini pemain akan memainkan satu karakter atau lebih yang kemudian karakter tersebut harus bertarung dengan karakter-karakter.

d. Shooting

Game shooting adalah tipe game yang sebenarnya cukup sederhana, karena pemain hanya cukup menembak musuh-musuh yang menghalangi selama pemain berlangsung.

e. RTS (*Real Time Strategy*)

Pada *game* bertipe RTS ini, sebagai pemain, melakukan kontrol terhadap satu atau lebih karakter untuk melakukan tindakan-tindakan tertentu guna menyelesaikan masalah (misi) di dalam game tersebut, sebagian besar game RTS adalah game strategy perang.

f. RPG (*Role Playing Game*)

Ini adalah salah satu tipe *game* yang dibuat dengan alur cerita penuh intrik yang begitu kental, pengembangan watak karakter secara mendalam.

g. Simulation

Sama seperti artinya yang berarti simulasi, di dalam *game* ini di berikan penggambaran konsep permainan dengan segala suatu hal di dalam kehidupan nyata.

h. Puzzle

Game jenis ini biasanya mengenai pemecahan sebuah teka-teki, seperti menyusun gambar, menyusun kata, menyamakan warna, melewati labirin, memecahkan perhitungan matematika dan lainnya. Contoh dari *game* jenis ini adalah Tetris, Minesweeper, Memory dan sebagainya.

i. Edukasi - *EduGames*

Video *Game* jenis ini dibuat dengan tujuan spesifik sebagai alat pendidikan, entah untuk belajr mengenal warna untuk balita, mengenal huruf dan angka, matematika, sampai belajar bahasa asing. Developer yang membuatnya, harus memperhitungkan berbagai hal agar *Game* ini benar-benar dapat mendidik, menambah pengetahuan dan meningkatkan ketrampilan yang memainkannya. Target segmentasi pemain harus pula disesuaikan dengan tingkat kesulitan dan design visual ataupun animasinya. Contoh edu*Game*s : Bobi Bola, Dora the explorer, Petualangan Billy dan Tracy.

2.2.4 Elemen Dasar *Game*

Elemen Dasar *Game* Menurut Teresa Dillon elemen-elemen dasar sebuah *game* [7] adalah :

1. Game Rule

Game rule merupakan aturan perintah, cara menjalankan, fungsi objek dan karakter di dunia permainan Dunia Game Dunia *game* bisa berupa pulau, dunia khayal, dan tempat-tempat lain yang sejenis yang dipakai sebagai setting tempat dalam permainan *game.*

2. Plot

Plot biasanya berisi informasi tentang hal-hal yang akan dilakukan oleh *player* dalam *game* dan secara detail ,perintah tentang hal yang harus dicapai dalam *game.*

3. Thema

Di dalam biasanya ada pesan moral yang akan disampaikan *Character* atau pemain sebagai karakter utama maupun karakter yang lain yang memiliki ciri dan sifat tertentu.

4. Object

Merupakan sebuah hal yang penting dan biasanya digunakan pemain untuk memecahkan masalah, adakalanya pemain harus punya keahlian dan pengetahuan untuk bisa memaninkannya.

5. *Text, grapich* dan *sound*

   Game biasanya merupakan kombinasi dari media teks, grafik maupun suara, walaupun tidak harus semuanya ada dalam permainan game.

6. Animasi

   Animasi ini selalu melekat pada dunia *game* , khususnya untuk gerakan karakter karakter yang ada dalam game, properti dari objek.

7. *User Interface*

   Merupakan fitur-fitur yang mengkomunikasikan user dengan game, agar user lebih mudah menggunakan game tersebut.

2.2.5 Metode Animasi

Animasi merupakan suatu teknik menampilkan gambar berurut sedemikian rupa sehingga penonton merasakan adanya ilusi gerakan (motion) pada gambar yang ditampilkan [6]. Animasi dapat dibuat dengan tiga teknik berbeda, yaitu *Image, Xoring,* dan *Make*, dan dalam pergerakannya dapat bertipe *object sprite* atau *object frame*, dan juga bisa dibedakan atas metode animasi yang digunakan antara animasi *frame*, *bibliting* dan *realtime*. Walaupun terbagi atas berbagai definisi berbeda, tapi dalam prakteknya teori-teori tersebut dapat digabungkan atau saling berhubungan sehingga tidak murni dipakai sendiri. Macam-macam animasi yang digunakan dalam membuat sebuah *Game* akan dijelaskan sebagai berikut [6]:

a. Animasi dengan Teknik *Image*

Animasi dengan teknik ini adalah menyimpan *image* sebagai sebuah *sprite* dalam memori yang kemudian akan ditampilkan di *background*-nya. Dalam teknik ini animasi yang disimpan harus berlatar belakang sesuai *background*-nya. Animasi dengan teknik ini biasanya sulit dalam pembuatan gambarnya, sebab harus banyak dan melakukan penyamaan dan posisi. Akan tetapi teknik ini mudah dalam hal memainkan animasinya.

b. Animasi dengan Teknik *Xoring*

Teknik ini adalah teknik animasi yang mudah dan sederhana, sebab selain gambarnya satu *sprite*, cara menampilkannya juga jauh lebih mudah

dibanding dengan teknik sebelumnya. Pembuatan gambarnya sangat mudah, sebab yang dibuat adalah *sprite*-nya saja dan tidak perlu menyamakan dengan *background*nya. Kelemahan dari teknik ini adalah memiliki efek buruk yaitu tembus pandang 24 dan mengganti warna *sprite*, maka hal ini tidak baik digunakan dalam animasi yang *background*-nya bergambar.

c. Animasi dengan Teknik *Make*

Animasi dengan teknik ini biasanya digunakan untuk animasi umum, tapi biasanya digunakan untuk proses pembuatan animasi. Animasi dengan teknik ini memiliki *sprite* yang terus menerus di generate oleh program, kemudian ditampilkan dengan perhitungan tertentu. Animasi ini biasanya dilakukan oleh 3D *modelling* dan *shading* oleh *software* seperti AutoCad, 3D Studio, Presidio 3D Workshop dan lain-lain. Operasi diatas haruslah di generate secara langsung dengan perhitungan sehingga saat pembuatan hampir bersamaan dengan saat menampilkannya.

d. Animasi dengan Tipe *Object Sprite*

Animasi ini menggunakan *sprite* sebagai pemeran utama sedangkan *object* lainnya hanya *background* diam. Prosesnya adalah mebuat gambar *sprite* dengan latar belakang warna hitam, lalu dibuat juga *sprite* yang sama tetapi berwarna hitam dan latar belakangnya adalah warna tertinggi, kemudian ditempatkan dengan pertama-tama menyimpan *background* yang akan ditimpa oleh *sprite* dan ditempatkan *sprite* dengan Xor dan Ditimpa dengan warna tertinggi secara Xor.

e. Animasi dengan Tipe *Object Frame*

Animasi ini menitik beratkan animasi yang dimainkan hanya pada *sprite object*-nya saja, akan tetapi seluruh *background*-nya juga seolah-olah ikut digerakkan.

f. Metode Animasi *Frame*

Metode ini adalah metode animasi yang mendukung tipe *object frame*. Karena metode animasi dengan metode *fullscreen*, maka *frame* yang tampil haruslah disiapkan terlebih dahulu dalam bebera page sebelumnya. Karena hal tersebut maka pengambilan gambarnya haruslah sangat cepat, sehingga tidak menjadikan animasi lamban dan tersendat. Animasi *frame* ini haruslah

menampilkan gambar *fullscreen* yang bergerak, agar efek tersendat dari pergantian *frame* tidak menyolok.

g. Metode Animasi *BitBlt* 25

Metode animasi ini biasanya disebut *sprite animation, array animation, blocked animation, partial screen animation, snapshot animation* atau *arcade animation*. Prinsip dari metode ini adalah menyimpan *image* dan memainkan animasinya dalam bentuk satu atau bebera *sprite* kecil. BitBlt ini digunakan untuk membuat program permainan dan program berbasis grafis terutama dalam membuat animasi sederhana.

h. Metode Animasi *Real-Time*

Dalam metode ini biasanya semua animasi yang sedang tampil atau yang akan dibuat dilakukan bersama sehingga tidak perlu disiapkan terlebih dahulu. Karena animasi ini lambat dan tersendat maka animasi dengan metode ini akan bagus jika pergerakan yang akan dilakukan adalah tidak diketahui sebelumnya dan tiba-tiba muncul. Metode ini tidak disarankan untuk animasi biasa-biasa saja, akan tetapi sebaiknya digunakan untuk keperluan khusus seperti *rotating* dan *tweening*.

2.2.6 Metode Getting Things Done (GTD)

Metode *Getting Things Done* (GTD) merupakan sebuah metode yang dirancang dengan menitikberatkan aspek kerja efektif dan kerja efisien, dibuktikan dengan implementasi pengelompokan daftar kerja dan penentuan eksekusi tugas yang diserahkan kembali ke individu dimana kendali penuh ada pada individu tersebut. Pada akhirnya, individu yang menerapkan metodologi GTD akan mendapati dirinya dapat menyelesaikan banyak pekerjaan dan lebih produktif dengan mendapati stress dan tekanan dari pekerjaannya sendiri jauh lebih sedikit [5].

Metodologi GTD menegaskan rutinitas-rutinitas yang dapat dilakukan seseorang dalam satu waktu, dan menciptakan kondisi kerja yang lebih memadai bagi pelakunya. [5] Inti dari metodologi ini adalah:

a. ***Collection*:** Mengumpulkan semua data, catatan pekerjaan, dan daftar pekerjaan yang ada dalam satu wadah dan tidak disortir.

b. ***Process*:** Penyortiran data yang telah dikumpulkan, dan menentukan aksi selanjutnya terhadap tiap data yang disortir.

   Jika data tersebut memerlukan aksi lanjutan, maka pelaku bisa:

   1. Langsung melakukannya saat itu juga jika bisa dilakukan dalam dua menit;
   2. Dicatat di *Monitor List* atau catatan pekerjaan yang akan dilakukan; atau
   3. Tunda pekerjaan tersebut dengan mencatatnya di *Action List* atau daftar pekerjaan yang bisa dikerjakan nanti.

   Jika data tersebut tidak memerlukan aksi lanjutan, maka pelaku harus menentukan apa data tersebut memiliki nilai nantinya. Jika data tersebut harus melalui pemikiran lebih jauh, maka masukkan ke *Incubate List* dan dicatat di *Sometime/Maybe List*, dan jika data tersebut memiliki nilai arsip, arsipkan di *Archive List*

   Jika data tersebut tidak memerlukan aksi lanjutan, tidak perlu pemikiran lebih lanjut dan tidak memiliki nilai referensi, maka masukkan data tersebut ke *Junk List*.

c. ***Organize***: Mengambil data yang telah disortir dan memasukannya ke dalam sebuah form harian untuk pengalokasian waktu pengerjaan tiap data.

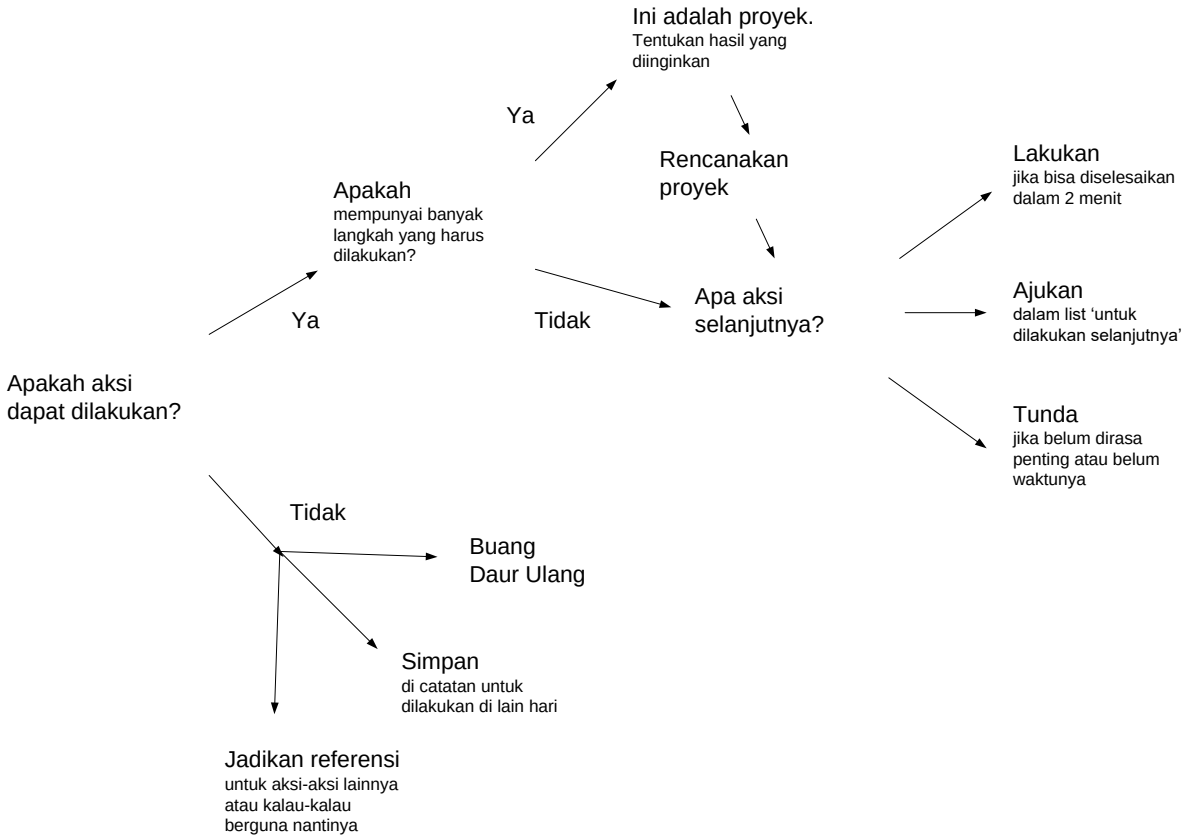
d. ***Review***: Melakukan kajian berkala untuk memastikan sistem yang dipakai bisa dipercaya, dengan mengkaji proses Collect dan Process kalau-kalau ada yang terlewat, mengkaji proyek apakah memiliki aksi lanjutan yang dapat diterima, memangkas daftar aksi-aksi yang tidak diperlukan, dan melihat kembali Sometime/Maybe List untuk data yang bisa dieksekusi selanjutnya.

e. ***Do***: Lakukan pekerjaan yang sudah disortir pada form yang sudah dibuat.

   Proses ini tergantung dan termotivasi oleh konsep-konsep terkait lainnya, juga tergantung situasi dan kondisi yang sedang terjadi.

2.2.6.1 Pemetaan GTD

Berdasarkan dasar metodologi, bisa digambarkan ‘aturan main’ individu dalam melakukan pekerjaan jika menerapkan metodologi GTD dalam merumuskan daftar kerja [5].

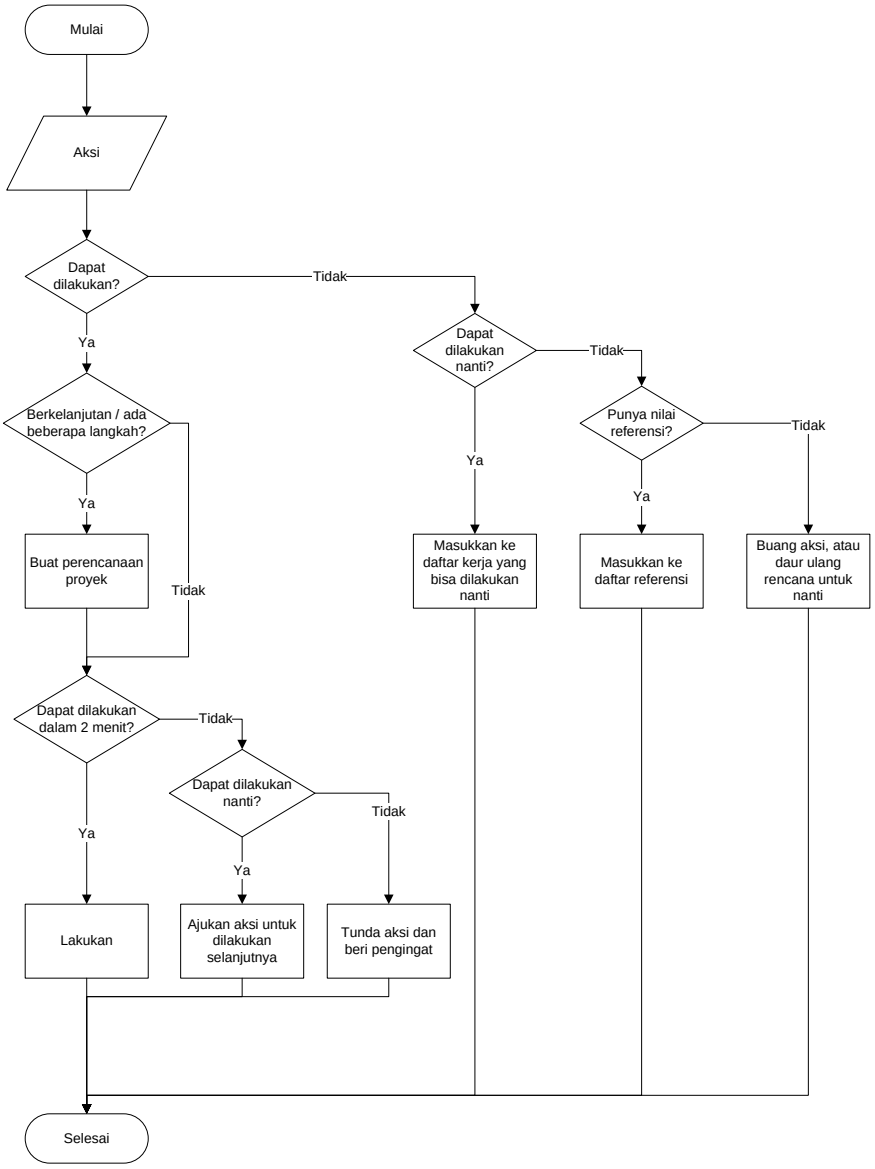


Gambar 2.2 Alur Metode GTD

Pada prakteknya, hasil yang dicapai akan berbeda-beda tiap individu tergantung seberapa besar kepentingan dan prioritas yang diterapkan oleh individu tersebut walaupun akhirnya terbentuk daftar kerja yang sesuai dengan keinginan individu tersebut dan dia dapat mengerjakannya dengan sedikit atau bahkan tanpa merasa stress berlebih.

2.2.6.2 Struktur GTD

Setelah melakukan studi dan pengamatan pada metodologi GTD, maka dapat dilakukan analisis bagaimana prosedur metodologi GTD dalam bentuk *flowchart*:



Gambar 2.3 Prosedur Metode GTD

Berikut adalah penjelasan dari *flowchart* metode GTD hasil analisis:

a. Aksi adalah satu rencana kegiatan yang akan dikerjakan oleh individu. Tiap aksi yang memungkinkan untuk dilakukan harus dicatat.
b. Jika aksi memungkinkan untuk dilakukan, tentukan apakah berkelanjutan atau tidak. Jika berkelanjutan atau memiliki beberapa langkah, buat perencanaan proyek dengan tajuk aksi tersebut.
c. Jika aksi tidak memungkinkan untuk dilakukan, cek apakah dapat dilakukan di lain hari atau memiliki nilai referensi. Jika dapat dilakukan di lain hari, masukkan ke daftar tunda yang bisa dikerjakan nanti. Jika memiliki nilai

referensi, masukkan ke daftar referensi. Jika tidak keduanya, buang atau daur ulang rencana aksi tersebut.

d. Jika aksi memungkinkan untuk dilakukan, cek apakah:
    1. Dapat dilakukan dalam dua menit, lakukan saat itu juga.
    2. Dapat dilakukan namun tidak dalam dua menit, ajukan dalam daftar kerja untuk dikerjakan nanti.
    3. Tidak dapat dilakukan saat itu juga, tun.

2.2.7 Manajemen Waktu

Management Waktu adalah sebuah perencanaan, pengorganisasian, penggerakan, dan pengawasan produktivitas waktu. Waktu menjadi salah satu sumber daya untuk segala aktivitas. Sumber daya yang harus dikelola secara efektif dan efisien. Efektifitas terlihat dari tercapainya tujuan menggunakan waktu yang telah ditetapkan sebelumnya. Dan efisien tidak lain mengandung dua makna,yaitu: makna pengurangan waktu yang ditentukan, dan makna investasi waktu menggunakan waktu yang ada [3] . Kualitas manajamen waktu berpedoman kepada empat indikator, yaitu :

a. Menetapkan Sasaran

Merencanakan waktu dengan cara menetapkan sasaran, sasaran yang hendak ingin dikerjakan yang terbagi dalam tujuan jangka pendek maupun jangka panjang. Menetapkan usaha pada dua atau tiga hal utama yang ingin diselesaikan daripada mengerjakannya sedikit demi sedikit. Sasaran yang spesifik terbagi dalam tujuan yang nyata dan wajar dapat dicapai.

b. Prioritas

Proses menentukan prioritas melibatkan perencanaan, penentuan prioritas sebaiknya ditetapkan setiap hari. Tuliskan semua yang hendak ingin dikerjakan besok kemudian golongkan menurut kepentingannya. Misalnya : “A”,”B” atau “C”. Mulailah dengan prioritas “A”, sehingga waktu yang digunakan untuk hal yang terpenting dahulu. Kembangkan kebiasaan untuk membuat rencana setiap hari. Tuliskan semua sasaran dan rencana pada daftar harian dan ikuti daftar tersebut. Hal ini adalah waktu yang tercepat untuk mengorganisis waktu. Perencanaan harian dari sub-sasaran merupakan hal yang esensial untuk

mengelola waktu dengan baik. Supaya perencanaan dari sasaran hari menjadi efektif, prioritas harus ditetapkan.

c. Komunikasi

Setelah kebiasaan dianalisis, sasaran diidentifikasi dan prioritaskan, hal ini tidak berarti bahwa sebuah organisasi atau bisnis secara otomatis memanfaatkan jam kerjanya dengan cara paling efisien. Masih ada persoalan mengkomunikasikan sasaran dan tujuan kepada anggota dengan cara sedemikikan rupa sehingga apa yang direncanakan, itulah yang dilaksanakan. Dalam kenyataannya, komunikasi menjadi alat manajemen untuk menyatukan kegiatan organisasi dengan mana sasaran perusahaan dapat dicapai. Sanagat penting untuk menentukan seberapa efektif alat ini beroperasi sebelum bersandar kepadanya.

d. Penundaan

Semua perencanaan yang mungkin dapat dikerjakan. Pernyataan sasarand an prioritas dapat ditulis. Analisis penggunaan waktu sekarang dan pemantapan program untuk memperbaiki kekurangan dpat diterapkan. Tetapi semua tidak berguna jika tidak dilaksanakan. Kegagalan ini dinamakan penundaan. Kebanyakan waktu yang terbuang dilakukan oleh kita, bukan orang lain. Jadi sebisa mungkin untuk mengendalikan waktu, mulailah dengan menyingkirkan tugas yang tidak menyenangkan sepagi mungkin.

Penundaan merupakan batu sandungan terbesar bagi hampir semua orang yang mencari perbaikan dalam memanfaatkan waktunya. Ada tiga sebab yang mengarah ke penundaan, diantaranya : tidak menyenangkan, proyek yang sulit dan keraguan.

2.2.8 Skala *Likert*

Skala *Likert* ditemukan dan dikembangkan oleh Rensis Likert pada tahun 1932 di mana skala *Likert* merupakan metode skala bipolar karena dapat digunakan untuk mengukur tanggapan positif ataupun negatif. Skala *Likert* digunakan untuk mengukur sikap, pendapat dan persepsi seseorang atau sekelompik orang tentang fenomena social [9]. Pada skala *Likert*, variabel yang akan diukur dijabarkan menjadi indikator variabel. Setiap jawaban dihubungkan dengan bentuk pernyataan

atau dukungan sikap yang diungkapkan dengan kata-kata misalnya untuk pernyataan positif [9].

Dalam melakukan perhitungan persentase terhadap responden, menggunakan rumus jumlah jawaban responded dikali dengan poin penilaian. Adapun cara menghitung poin dalam skala *Likert*, dapat dilihat pada tabel 2.21.

Tabel 2.1 Poin Penilaian Skala *Likert* [9]

Jawaban	Poin Penilaian
Sangat Setuju	5
Setuju	4
Ragu-ragu	3
Tidak Setuju	2
Sangat Tidak Setuju	1

2.2.9 *Game Engine*

Pada tugas akhir ini, Aplikasi yang digunakan ialah *Rpg Maker VXAce* sebagai *game engine. Rpg Maker VXAce* suatu *game engine* yang terus berkembang. Penggunaan *engine* versi *lite* dibatasi dengan beberapa fitur yang dikurangi atau bonus modul / prefab tertentu yang ditiadakan dan hanya tersedia untuk pengguna berbayar.

Rpg Maker VXAce pada dasarnya adalah game engine dengan 2 dimensi tampilan *top down*, merupakan sebuah tampilan yang sudut pandangnya dilihat dari atas. Keunggulan Rpg Maker VX Ace adalah memiliki kemampuan membuat game dengan genre *RPG* menjadi lebih mudah, dikarenakan engine ini sudah diseting secara default untuk pembuatan game *RPG*. Meskipun begitu tidak menutup kemungkinan *engine* ini dapat membuat game dengan genre lainnya. Engine ini memiliki beberapa fitur diantaranya :

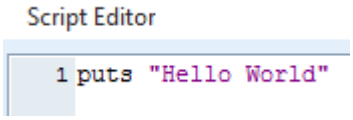
2.2.9.1 *Point-and-click engines*

Engine ini merupakan *engine* yang sangat dibatasi, tapi dibuat dengan sangat *user friendly*. Pengguna bahkan bisa mulai membuat game sendiri menggunakan *engine* seperti *GameMaker, Torque Game Builder* dan *Unity3D*. Dengan sedikit memanfaatkan coding, sudah bisa merilis *game point-and-click* yang kamu banget. Kekurangannya terletak pada terbatasnya

jenis interaksi yang bisa dilakukan dan biasanya hal ini mencakup semuanya, mulai dari grafis hingga tata suara [7].

2.2.9.2 *Ruby Gaming Scripts System* 3 (RGSS3)

Ruby Gaming Scripts System 3 atau yang sering disebut sebagai *RGGS3,* merupakan sebuah bahasa pemrograman yang dimiliki oleh engine rpgmaker vx ace, scripts ini merupakan pengembangan dari bahasa pemrograman *ruby.* Dalam rpgmaker vx ace scripts digunakan untuk mengcustomisasi beberapa bagian dari sistem standar dari rpgmaker vx ace, misalnya : battle system, ui, dan sebagainya. Contoh penulisan RGSS3 :



Gambar 2.4 Contoh Penulisan RGSS3

2.2.9.3 *Run Time Packed* (RTP)

Run Time Packed atau yang sering disebut sebagai RTP, merupakan sebuah paket untuk menyediakan asset untuk pembuatan game pada engine ini. Hal ini tentunya dapat mempermudah pengguna apabila mereka belum memiliki tim untuk pembuatan assetnya.

2.2.10 *Object Oriented Analysis Design*

Konsep OOAD mencakup analisis dan desain sebuah sistem dengan pendekatan objek, yaitu analisis berorientasi objek (OOA) dan desain berorientasi objek (OOD). OOA adalah metode analisis yang memeriksa requirement (syarat/keperluan) yang harus dipenuhi sebuah sistem dari sudut pandang kelas-kelas dan objek-objek yang ditemui dalam ruang lingkup sistem. Sedangkan OOD adalah metode untuk mengarahkan arsitektur software yang didasarkan pada manipulasi objek-objek sistem atau subsistem [8].

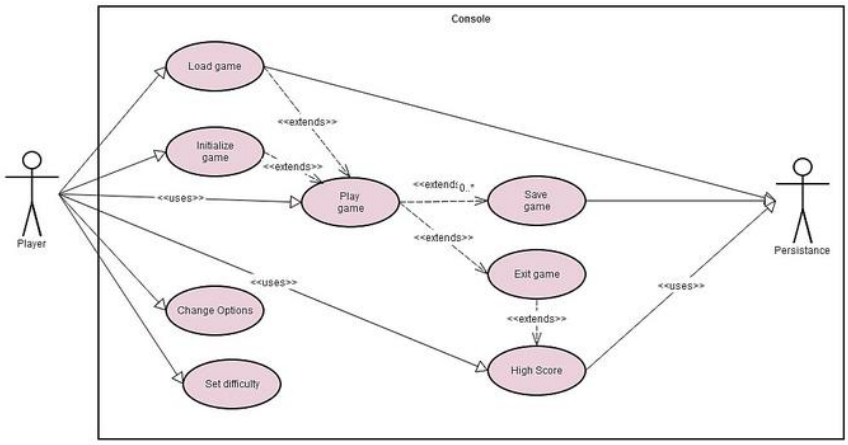
2.2.10.1*Unified Modeling Language* (UML)

Unified Modeling Language (UML) adalah termasuk ke dalam rumpun jenis pemodelan notasi grafis yang didukung oleh model-model tunggal. Pemodelan ini berguna untuk membantu dalam menjelaskan data rancang perangkat lunak yang dibangun dengan *object-oriented* (OO). UML merupakan standar terbuka yang

dikelola *Open Management Group* (OMG) yang berada dibawah naungan perusahaan-perusahaan konsorium terbuka. UML merupakan suatu bahasa pemodelan yang terdiri banyak model diantaranya adalah [8]:

a. Use Case Diagram

Diagram *use case* merupakan pemodelan untuk menggambarkan kelakuan (*behavior*) sistem secara keseluran yang akan dibuat.



Gambar 2.4 Use Case Diagram

Diagram *use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem yang akan dibuat. Dengan pengertian yang cepat, diagram *use case* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sebuah sistem dan siapa saja yang berhak menggunakan fungsi-fungsi tersebut. Yang ditekankan pada diagram ini adalah "apa" yang diperbuat sistem, dan bukan "bagaimana". *Use case* menjelaskan secara sederhana fungsi sistem dari sudut pandang user. Adapun komponen-komponen dalam use case diagram anataranya [8]:

1) *Actor*

Actor adalah segala hal diluar sistem yang akan menggunakan sistem tersebut untuk melakukan sesuatu. Bisa merupakan manusia, sistem, atau *device* yang memiliki peranan dalam keberhasilan operasi dari system.

2) *Use Case*

Mengidentifikasi fitur kunci dari sistem. Tanpa fitur ini, sistem tidak akan memenuhi permintaan *user/actor*. Setiap *use case* mengekspresikan *goal* dari sistem yang harus dicapai. Diberi nama sesuai dengan *goal*-nya dan digambarkan dengan *elips* dengan nama di dalamnya. Fokus tetap

pada *goal* bukan bagaimana mengimplementasikannya walaupun *use case* berimplikasi pada prosesnya nanti. Setiap *use case* biasanya memiliki *trigger*/pemicu yang menyebabkan *use case.*

3) *System*

Menyatakan batasan sistem dalam relasi dengan *actor-actor* yang menggunakannya (di luar sistem) dan fitur-fitur yang harus disediakan (dalam sistem). Digambarkan dengan segi empat yang membatasi semua *use case* dalam sistem terhadap pihak mana sistem akan berinteraksi. Sistem disertai label yang menyebutkan nama dari sistem, tapi umumnya tidak digambarkan karena tidak terlalu memberi arti tambahan pada diagram.

4) *Association*

Mengidentifikasikan interaksi antara setiap *actor* tertentu dengan setiap *use case* tertentu. Digambarkan sebagai garis antara *actor* terhadap *use case* yang bersangkutan. Asosiasi bisa berarah (garis dengan anak panah) jika komunikasi satu arah, namun umumnya terjadi kedua arah (tanpa anak panah) karena selalu diperlukan demikian.

5) *Dependency*

Dependency <<include>>

a. Mengidentifikasi hubungan antar dua *use case* di mana yang satu memanggil yang lain.
b. Jika pada beberapa *use case* terdapat bagian yang memiliki aktivitas yang sama maka bagian aktivitas tersebut biasanya dijadikan *use case* tersendiri dengan relasi dependensi setiap *use case* semula ke *use case* yang baru ini sehingga memudahkan pemeliharaan.
c. Digambarkan dengan garis putus-putus bermata panah dengan notasi <<include>> pada garis.
d. Arah mata panah sesuai dengan arah pemanggilan.

Dependency <<extend>>

a. Jika pemanggilan memerlukan adanya kondisi tertentu maka berlaku dependensi <<extend>>.

b. Digambarkan serupa dengan dependensi <<include>> kecuali arah panah berlawanan.

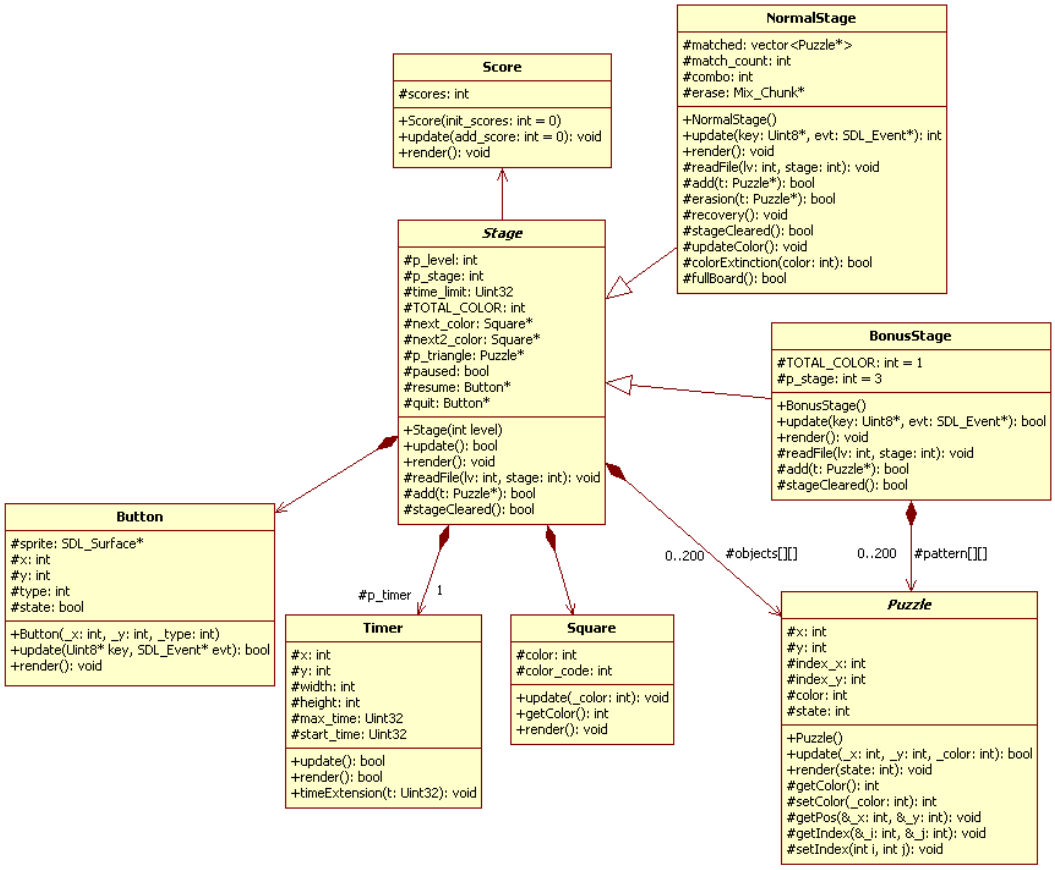
c. Note: konsep "*extend*" ini berbeda dengan "extend" dalam Java.

6) *Generalization*

Mendefinisikan relasi antara dua *actor* atau dua *use case* yang mana salah satunya meng-*inherit* dan menambahkan atau *override* sifat dari yang lainnya. Penggambaran menggunakan garis bermata panah kosong dari yang meng-*inherit* mengarah ke yang di-*inherit.*

b. Class Diagram

Class diagram merupakan diagram yang selalu ada di pemodelan system berorientasi objek.



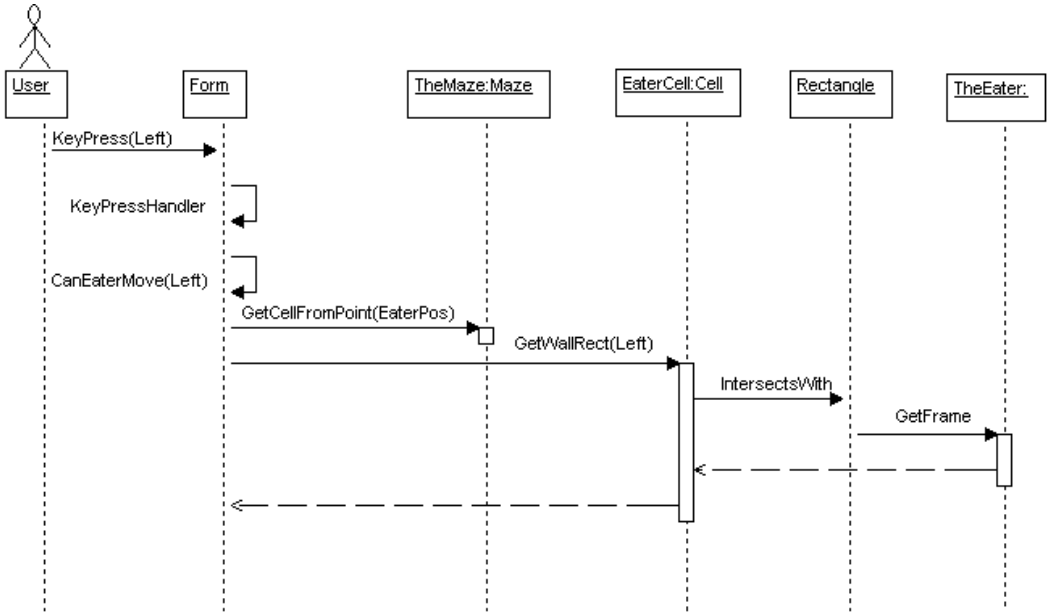
Gambar 2.5 Class Diagram

Class diagram menunjukkan hubungan antar class dalam system yang sedang dibangun dan bagaimana mereka saling berkolaborasi untuk mencapai satu tujuan. Kelas pada kelas diagram terdiri dari 3 bagian utama yaitu nama kelas, isi property dari kelas beserta metode yang ada pada kelas tersebut. Kelas

juga memiliki jenis-jenis hubungan seperti asosiatif, dependensi, agregasi, komposisi, spesifikasi dan generalisasi. Hubungan ini digunakan untuk menggambarkan bagaimana hubungan dan interaksi yang terjadi antar kelas. Masing-masing komponen penyusun kelas memiliki hak akses seperti *public*, *private* dan *protected* [8].

c. Sequence Diagram

Sequence diagram menjelaskan secara detail urutan proses yang dilakukan dalam system untuk mencapai tujuan dari *use case*.



Gambar 2.5 *Sequence Diagram*

Interaksi yang terjadi antar *class*, operasi apa saja yang terlibat, urutan antar operasi dan informasi yang diperlukan oleh masing-masing operasi [8].

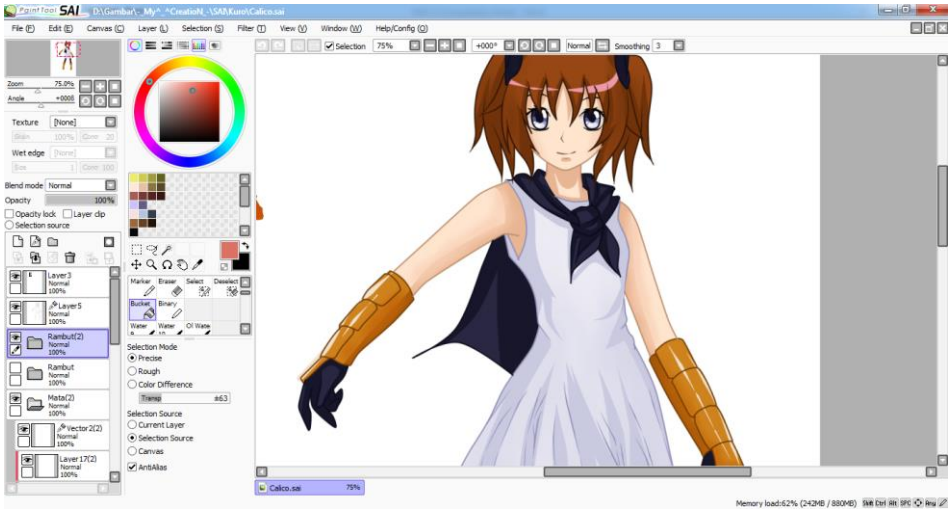
d. Activity Diagram

Activity diagrams menggambarkan berbagai alir aktivitas dalam sistem yang sedang dirancang, bagaimana masing-masing alir berawal, decision yang mungkin terjadi, dan bagaimana mereka berakhir. Activity diagram juga dapat menggambarkan proses paralel yang mungkin terjadi pada beberapa eksekusi. Activity diagram merupakan state diagram khusus, di mana sebagian besar state adalah action dan sebagian besar transisi di-trigger oleh selesainya state sebelumnya (internal processing). Oleh karena itu activity diagram tidak menggambarkan behaviour internal sebuah sistem (dan interaksi antar

subsistem) secara eksak, tetapi lebih menggambarkan proses-proses dan jalur-jalur aktivitas dari level atas secara umum. Menggambarkan proses bisnis dan urutan aktivitas dalam sebuah proses. Dipakai pada business modeling untuk memperlihatkan urutan aktivitas proses bisnis. Struktur diagram ini mirip flowchart atau Data Flow Diagram pada perancangan terstruktur. Sangat bermanfaat apabila kita membuat diagram ini terlebih dahulu dalam memodelkan sebuah proses untuk membantu memahami proses secara keseluruhan. Activity diagram dibuat berdasarkan sebuah atau beberapa use case pada use case diagram [8].

2.2.11 Paintool SAI

Paint tool sai adalah sebuah software desain buatan jepang yang berfungsi sebagai untuk menggambar dan mewarnai, software ini sering digunakan seorang pembuat komik di Jepang.



Gambar 2.6 Paintool SAI

Karena penggunaan softwarenya yang relatif lebih ringan dari software desain lainnya. Selain itu software ini juga memiliki fitur meruncingkan garis untuk pengguna mouse.