BAB II

TINJAUAN PUSTAKA

2.1. Teori Dasar

Teori dasar akan menjelaskan teori utama yang dibutuhkan untuk mendukung penulis dalam judul Implementasi *Finite State Machine* Dalam *Game* Edukasi Bahasa Jepang. Penulis akan menguraikan teori mengenai *Android, Game, dan Finite State Machine* dengan menggunakan beberapa jurnal sebagai pedoman pembuatan laporan ini.

2.1.1. *Game*

Suatu hal yang dimainkan dengan tujuan untuk menghibur atau mengisi waktu luang. Permainan tersebut biasanya memiliki aturan- aturan dengan hasil yang sudah terukur, sehingga jika hasilnya berbeda, maka nilai yang diberikan juga akan berbeda (Janata, Thyo Priandika, & Gunawan, 2022). Bermain *game* juga dapat mengasah otak karena peraturan-peraturan yang terdapat dalam *game* tersebut membuat permainan yang dimainkan lebih menantang karena pemain harus menggunakan strategi yang pola pikir *problem solvin*g yang baik untuk menyelesaikan *game* tersebut.

Berdasarkan platformnya, *game* dibagi menjadi beberapa jenis (Abidzar Tawakal, 2021), antara lain:

1. ***Console Games***

Merupakan sebuah permainan yang hanya bisa dimainkan dalam mesin yang sudah dirancang secara khusus untuk permainan tersebut. Mesin ini biasanya

memerlukan sebuah *monitor* untuk menampilkan video permainan yang dimainkan, dan juga *controller* sebagai alat untuk mengendalikan.

2. *Handheld Games*

Sama halnya dengan *console games*, *handheld games* juga memerlukan sebuah mesin khusus untuk bisa memainkan permainannya, namun mesin yang digunakan memiliki ukuran yang lebih kecil, sehingga *console handheld* dapat digenggam dengan tangan dan dibawa ke mana-mana, itulah mengapa disebut *handheld* (dapat digenggam). Menjadi salah satu terobosan terbaik dalam pasar perusahaan elektronik dalam bidang *game*.

3. *PC Games*

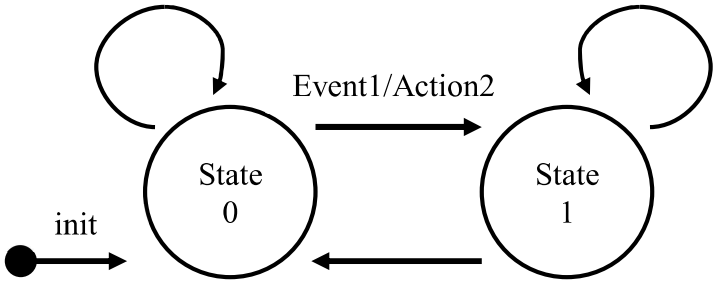
Permainan ini memerlukan mesin komputer, dan bukan mesin console. Untuk menjalankan permainan tersebut pun, komputer memerlukan sebuah unit pemrosesan grafis yang dinamakan Direct3D, terutama permainan yang menggunakan grafis tiga dimensi. *PC games* juga menjadi salah satu alasan utama *graphics card* (kartu grafis) terus berkembang dan di pasarkan.

4. *Mobile Games*

Permainan yang dimainkan dengan *platform mobile*, seperti handphone. Memiliki layar monitor yang relatif kecil, sehingga jika harus dibandingkan kualitas grafisnya, *Mobile Games* mungkin tidak bisa lebih unggul dari platform lainnya. Namun jika dibandingkan dari sisi portabilitasnya, perangkat mobile akan jauh lebih unggul karena tidak membutuhkan kabel maupun perangkat tambahan seperti monitor, terlebih lagi *handphone* juga sudah dimiliki orang sebagian besar orang.

2.1.2. *Finite State Machine*

Sebuah mesin abstrak, yang menggambarkan tingkah laku system dengan menggunakan 3 hal yaitu, *state* (keadaan), *event* (kejadian), dan *action* (aksi). Mesin ini dapat merubah *state* satu ke *state* lainnya sebagai *output* dari sebuah masukan, baik dari perangkat luar maupun komponen *system* itu sendiri, namun mesin ini hanya dapat berada pada satu *state* pada satu waktu saja. Perubahan tersebut dinamakan *transtition* (Asrianda & Zulfadli, 2022).



Gambar 2.1 *Finite State Machine*
Sumber Gambar: Google

Pembuatan flowchart dan diagram lainnya menggunakan finite state machine sebagai dasarnya, sering digunakan sebagai metode (Handoko & Aranski, 2019), mesin *finite state* terdapat dalam perangkat yang ditemukan sehari-hari, mesin pintu putar otomatis yang biasa ditemukan pada stasiun. Mesin pintu putar hanya memiliki dua *state* (keadaan), yaitu "terkunci", dan "terbuka". Kemungkinan *input* (masukan) yang diberikan yaitu, "memasukkan koin", atau "mendorong". Jika mesin tersebut dalam keadaan terkunci, maka perintah "mendorong", tidak akan mengubah keadaan mesin tersebut, namun jika diberikan perintah "memasukkan koin", maka keadaan mesin akan berubah menjadi "terbuka".

2.1.3. *Android*

Sistem operasi *mobile* berbasis Linux yang mencangkup *OS*, *middleware*, dan *software* (Siregar & Handoko, 2022). Para pengembang dan pembuat aplikasi dapat bebas berkarya dan berkembang karena *Android* merupakan platform yang terbuka (*open source*). Melalui *Google Play*, para pengguna *Android* dapat mengunduh banyak sekali aplikasi sesuai kebutuhannya.

Open Handset Alliance, dimana kerja sama yang terbentuk dari 34 perusahaan telekomunikasi, termasuk *Google*, berkumpul untuk mengembangkan *Android* yang merupakan pendatang baru piranti lunak untuk ponsel. *Android* bersama *Open Handset Alliance* melakukan perilisan perdananya pada tanggal 5 November 2007. Para developer diberikan kebebasan dalam mengembangkan aplikasinya karena *Android* juga merupakan sistem operasi *Open Source* (Anam, Emerlada, Erlinda, Tashid, & Nasution, 2023).

Sejak perilisan perdana, *Android* mengalami perkembangan versi secara rutin hingga saat ini. Dengan terus diperbaharuinya sistem *Android*, membuatnya menjadi tetap relevan dalam perkembangan teknologi. Berikut adalah perkembangan versi *Android* sejak awal perilisan antara lain:

a. Android 1.0 Astro (2008)

Versi pertama *Android* yang diresmikan pada tanggal 23 September 2008. Pada awalnya versi ini memiliki nama Astro, namun nama tersebut tidak dilanjutkan karena terdapat masalah dalam hak cipta. Fitur dari versi *Android* ini adalah tersedianya tempat bernama *Android Market*, dimana pengguna dapat mengunduh berbagai aplikasi.

b. *Android 1.1 Bender (2009)*

Sebuah upgrade dari versi sebelumnya, dari sisi performa dan UI yang lebih baik. Namun sama seperti versi terdahulu, terdapat masalah dalam hak cipta pada pemberian nama *Bender*, sehingga nama tersebut tidak digunakan.

c. *Android 1.5 Cupcake (2009)*

Awal dari pemberian nama makanan manis kepada versi-versi *Android* yang akan datang, dan semua nama tersebut akan mengikuti inisial dari urutan versi yang bersangkutan. Versi 1.5 adalah yang ketiga dari *Android*, sehingga ia di beri nama yang memiliki inisial C, yaitu *Cupcake*. Versi ini menawarkan aplikasi yang beragam, mulai dari kamus, mengolah *video*, *widget*, dan pengunggah *video* ke *Youtube*.

d. *Android 1.6 Donut (2009)*

Dilanjutkan dengan versi keempat yaitu *Donut*, yang dirilis masih pada tahun yang sama 2009. Fitur jaringan seperti *CDMA* dan *EDVO* ditambahkan dalam versi ini. Selain itu juga terdapat pengembangan dalam sistem kamera, persentase baterai dan fitur *search engine* pada *Android market*.

e. *Android 2.0 Éclair (2009)*

Android Éclair memperkenalkan fitur GPS ke dalam sistemnya. Kini memiliki fitur *flash* dan *zoom* pada kameranya, fitur *multitouch*, *live wallpaper*, dan layanan *Microsoft Exchange*. Pengguna juga dapat mencari semua pesan *SMS* yang sudah tersimpan dalam perangkat. Selain itu *Android Éclair* juga didukung untuk *HTML5*, memiliki UI *browser* yang lebih baik dan fitur *zoom* dengan melakukan *double-tap*.

f. *Android 2.2 Froyo (2010)*

Berasal dari singkatan, nama Froyo berasal dari *Frozen Yogurt*. Versi ini mulai meluas ke pabrik-pabrik *smartphone*. Perubahan siginifikan yang terdapat dalam versi ini adalah terdapatnya fitur *USB* tethering dan *WIFI hotspot*. Selain itu *Android* Froyo bisa menggunakan *Adobe Flash Player*, memungkinkan pengguna untuk merekam video.

g. *Android 2.3 Gingerbread (2011)*

Versi yang membawa *Android* menjadi sistem operasi *mobile* yang mendunia. Fitur utama yang ditambahkan dalam versi ini adalah sistem dapat melakukan *video call* dengan kamera depan. Versi ini juga dibuat khusus untuk membuat berbagai *game* dan aplikasi dalam *Google Play* menjadi lebih *compatible* dan optimal.

h. *Android 3.0 Honeycomb (2011)*

Dikhususkan untuk pengguna tablet, versi ini menawarkan tampilan yang mempermudah penggunanya untuk mengeksplor, tethering *Bluetooth*, melakukan *video chat*, dan melakukan *multitasking* dengan perpindahan antar aplikasi yang sedang berjalan di *background*. Tidak hanya itu *Android Honeycomb* juga memungkinkan pengguna untuk memasuki mode layar penuh ketika dalam galeri foto.

i. *Android 4.0 Ice Cream Sandwich (2012)*

Versi ini mengoptimalisasi kelemahan dari versi sebelumnya, dimana pada versi *Honeycomb*, fiturnya hanya bisa dinikmati oleh pengguna tablet. Kini fitur tersebut sudah terintegrasi ke dalam *smartphone* pada umumnya, dengan tambahan UI dan *widget*.

j. *Android 4.1 Jellybean (2012)*

Penambahan performa dari versi sebelumnya, *Android Jellybean* memberikan kostumisasi *widget*, dan UI yang semakin mulus dan beragam. Fokusnya pada versi ini adalah kostumisasi *keyboard* dan adanya ramalan cuaca dan lalu lintas dalam pencarian *Google*.

k. *Android 4.4 KitKat (2013)*

Memiliki fitur *emoji* dalam keyboardnya yang tersambung ke *Google Cloud Print*, membawa sensor langkah dan *batching*, dilakukannya optimalisasi performa untuk perangkat berspesifikasi rendah, dan memperkenalkan *status bar* yang transparan. Penghapusan aplikasi pesan dan *Movie Studio*, dan diganti dengan *Google Hangouts* yang mendukung fitur *SMS*.

l. *Android 5.0 Lollipop (2014)*

Telah menerapkan prosesor 64 bit, *Android Lollipop* memiliki daya tahan baterai yang lebih baik dikarenakan ia mampu melakukan optimalisasi penggunaan baterai. Terdapat perubahan *UI*, *pop up notification*, mode prioritas, dan mode *sleep*. Versi ini tidak hanya dapat dijalankan pada *smartphone*, namun juga pada *Android TV* dan *Google Fit*.

m. *Android 6.0 Marshmallow (2015)*

Membuat model izin yang sudah didesain ulang, dimana aplikasi tidak diberikan akses untuk segalanya secara otomatis. Ditambahkannya sensor pembaca sidik jari. Untuk mengurangi aktivitas aplikasi pada *background* perangkat juga diperkenalkan sistem *Doze*, sehingga dapat mengurangi penggunaan daya baterai. Selain itu juga terdapat penggunaan USB *type* C, dan mode 4K.

n. *Android 7.0 Nougat (2016)*

Peningkatan performa *UI*, lebih intuitif dengan mode *multi window*, fitur memilih warna kulit pada emoji *keyboard*, mode malam, kemampuan untuk menghapus beberapa aplikasi sekaligus, dan sistem *Doze* yang sudah di *upgrade* sehingga menjadi lebih efisien.

o. *Android 8.0 Oreo (2017)*

Dirilis pada tanggal 21 Agustus 2017, *Android* menambahkan fitur *Autofill* yang memudahkan pengguna untuk mengisi data data yang sudah pernah tersimpan sebelumnya. Versi ini juga memiliki kemampuan untuk membuka lebih dari satu aplikasi dalam satu layar. Penyederhanaan *UI* menjadi lebih simpel, dan adanya *notification dots*, yang memudahkan pengguna untuk mengetahui jika ada notifikasi.

p. *Android 9.0 Pie (2018)*

Mulainya teknologi *AI* diintegrasikan ke dalam sistem *Android*. Performa menjadi lebih cepat, dengan penggunaan baterai yang lebih hemat. Ditambahkannya fitur *adaptive brightness* yang menyesuaikan kecerahan berdasarkan apa yang ditampilkan layar, dan *bezel less*.

q. *Android 10 Q (2019)*

Memiliki aksesibillitas yang lebih baik dengan adanya fitur teks langsung dalam video. Versi ini berfokus pada penyempurnaan fitur mode gelap, yang berguna untuk menghindari layar menjadi terlalu silau, perubahan tampilan notifikasi menjadi bentuk *bubble*, dan penambahan fitur *sound amplifier*. *Android* Q merupakan nama yang muncul ketika sedang dalam pengembangan, namun

akhirnya diganti menjadi *Android* 10 yang menunjukan bahwa pada tahun 2019, *Android* berumur satu dekade.

r. *Android 11 Red Velvet Cake (2020)*

Berisi *API* yang dapat mendeteksi keberadaan koneksi jaringan 5G. Pengembangan ini juga mampu diterapkan ke perangkat yang memiliki layar berengsel seperti *smartphone* lipat. Peningkatan kinerja, privasi, keamanan, penyimpanan RAM, dan pengenalan izin "satu kali" ketika aplikasi ingin menggunakan kamera, mikrofon, dan lokasi.

s. *Android 12 Snow Cone (2021)*

Menawarkan fitur tampilan UI yang baru dengan *Material You*, dimana tampilan *widget* dapat diatur oleh penggunanya. Terdapat peningkatan dalam segi keamanan juga dalam versi ini, dimana aplikasi yang ingin meminta data lokasi kini hanya terbatas pada perkiraan lokasinya saja dibandingkan dengan titik lokasi tepatnya. *Android* 12 juga menambahkan fitur *spatial audio* dan MPEG-H 3D *Audio*.

t. *Android 13 Tiramisu (2022)*

Fitur media picker, dimana memungkinkan pengguna untuk hanya memilih foto dan video tertentu yang dapat diakses oleh aplikasi yang bersangkutan. Kompatibel dengan *Bluetooth* LE *Audio* dan LC3 *Audio Codec*, yang memungkinkan pengguna untuk menerima dan membagi data suara dengan beberapa perangkat sekaligus.

2.1.4. Bahasa Jepang

Bahasa yang secara resmi digunakan oleh negara Jepang untuk berkomunikasi, memiliki hubungan kemiripan pengucapan dan huruf yang digunakan pada negara tetangganya, yang tidak lain dari Cina. Pada abad ke-4, negara Cina membawa *Hanzi* yang merupakan bahasa mereka ke Jepang, meskipun memiliki bunyi yang berbeda, *Hanzi* inilah yang melandasi huruf-huruf *Kanji* yang sekarang ada dalam Bahasa Jepang, yang kemudian mengalami penyederhanaan aksara seiring waktu, menghasilkan huruf-huruf dasar yang bernama *Hiragana* dan *Katakana* (Sunarti, R.Y, & Damhudi, 2016).

a. Hiragana

Melambangkan kata-kata yang berasal dari Jepang asli, berisi suku kata tunggal, terbentuk dari penyederhanaan huruf *Kanji* yang kemudian dimodifikasi, sehingga memiliki bentuk melengkung. Digunakan pada jaman *Edo* (1603-1868), dan pada awalnya hanya digunakan oleh kaum wanita, sehingga hiragana dikenal dengan sebutan huruf wanita, dan dalam Bahasa Jepang itu sendiri disebut *onnade*.

b. Katakana

Sama seperti hiragana, katakana juga melambangkan suku kata tunggal, namun memiliki bentuk dan fungsi yang berbeda dengan huruf *hiragana*. Biasanya digunakan untuk kata-kata yang berasal dari bahasa asing, atau penekanan dari suatu kata. Berbeda dari *hiragana*, *katakana* berasal dari kanji yang dimodifikasi dengan cara mengambil salah satu bagian dari *kanji*, sehingga huruf *katakana* terkesan lebih kaku karena bentuknya tidak melengkung, dan bersudut tajam, membuat *katakana* sendiri disebut sebagai huruf laki-laki.

2.2. Teori Khusus

Teori khusus akan menjelaskan secara khusus teori yang mendukung penulis dalam judul Implementasi *Finite State Machine* Dalam *Game* Edukasi Bahasa Jepang. Penulis akan menguraikan teori-teori mengenai aplikasi pendukung yang digunakan, konsep *game* yang dirancang, serta penjelasan mengenai UML dengan menggunakan beberapa jurnal sebagai pedoman pembuatan laporan ini.

2.2.1. *Visual Novel*

Salah satu jenis *game* bergaya *anime* yang berfokus lebih kepada pembawaan cerita dan memproyeksikan novel menjadi bentuk gambar tidak bergerak yang dilengkapi dengan kotak teks untuk menyampaikan percakapan karakter dan narasi dari cerita yang bawakan. Umumnya *game* ini dimainkan untuk konsol seperti PSP, NDS, dan Switch. Secara *gameplay*, *visual novel* tidak menyajikan banyak hal selain karakter, teks dan *background*, namun karena memiliki tambahan aspek musik, pengalaman bermain *game* dapat menjadi lebih dramatis (Naratama, Prasida, & Prestiliano, 2023).

Game visual novel secara umumnya memiliki format *story-telling*, dimana pemain akan diceritakan sebuah cerita melalui dialog percakapan antar karakter, dapat dimainkan dengan santai, karena selain dapat melakukan *fast forward* untuk mempercepat cerita, pemain juga dapat melakukan *save* dan *load* kapanpun yang diinginkan. Pemain juga memiliki kemampuan untuk mengubah alur cerita melalui pilihan jawaban yang disediakan oleh *game* tersebut. Alhasil dalam sebuah *game visual novel*, terdapat lebih dari satu *ending* atau akhir cerita, baik itu akhir yang bahagia, maupun menyedihkan. (Anggraini & Fu, 2021).

2.2.2. *Visual Studio Code*

Sebuah *software* ringan dan handal dari *Microsoft* yang dibuat dengan fungsi *text editor*, mendukung berbagai macam bahasa pemrograman, kompatibel dengan sistem operasi *Windows*, *Mac*, dan *Linux*, serta dapat memasukkan bahasa pemrograman tambahan melalui *extension* dan *plugin* (Ningsih & Aruan, 2022).

Tidak hanya gratis, *visual studio code* juga sangat ringan dan cepat, memungkinkan pengguna untuk menggunakannya pada berbagai macam perangkat. Memiliki fitur *debugging* yang sangat baik, membuat pengguna dapat melihat letak kesalahan dalam *coding* dengan mudah, dan dengan adanya fitur *extension marketplace*, pengguna juga tidak perlu mencari plugin tambahan dari luar, dan memasukkannya kedalam aplikasi, cukup dengan menggunakan fitur ini saja pengguna dapat mengunduh *plugin* yang dibutuhkan secara cepat dan mudah.

2.2.3. Ren'Py

Sebuah *game engine* yang berfokus kepada pembuatan *game visual novel*, menggunakan bahasa pemrograman *python*, dan dapat merancang *game* yang dapat didistribusikan ke produk pengguna sistem operasi *Android*, *iOS*, dan *Windows*. Nama Ren'Py itu sendiri diambil dari kata "Ren'ai" yang mengartikan 'kisah cinta', dan Py dari *python*. Meskipun gratis dan *open source*, Ren'Py menawarkan fitur-fitur yang sangat menarik seperti kemampuan untuk membuat cerita non-linear atau bercabang, melakukan *save* dan *load game*, memberikan transisi dialog, dan fitur untuk menambahkan 'konten tambahan' (DLC) (Asyahda, Purno, & Wibowo, 2023).

2.2.4. *Paint Tool* SAI

Program yang dikembangkan *Systemax Software* untuk membuat gambar atau grafis ringan, dirilis pertama kali pada tanggal 25 Febuari 2008 (Napitupulu, 2021). Selain ringan, *tools* yang diberikan sangat mudah dimengerti, memudahkan pemula yang baru belajar menggunakannya. Namun SAI memiliki kekurangan dimana ketika gambar yang sudah didesain diubah ukurannya, kualitas gambar tersebut dapat menjadi pecah, selain itu kurangnya efek khusus juga menjadi pertimbangan ketika pengguna ingin membuat sebuah grafis yang sedikit rumit.

2.2.5. *Reaper*

Sebuah *software* yang digunakan untuk melakukan proses *audio production*, mulai dari *recording*, *mixing*, hingga *mastering*, dan termasuk DAW (*Digital Audio Workstation*) yang sangat lengkap dan fleksibel. *Software* ini tidaklah gratis, namun pengguna dapat mencoba *Reaper* dan menggunakan semua fiturnya secara lengkap tanpa ada batasan waktu, namun setiap membuka aplikasi, pengguna akan diminta untuk membeli lisensi untuk *Reaper* tersebut dan jika masih ingin mencobanya secara gratis, maka harus menunggu beberapa detik agar bisa mulai menggunakannya.

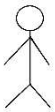
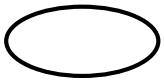
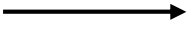
2.2.6. UML (Unified Modeling Language)

Pada tahun 1997, *Object Management Group* menciptakan sebuah metode visual dalam melakukan pemodelan perancangan sistem berorientasi objek yang dinamakan UML (*Unified Modelling Language*). Metode pengembangan yang berorientasi objek (Kesuma & Handoko, 2021), UML dapat diumpamakan sebagai *blueprint* dari sebuah *software*, berguna untuk memberikan sebuah gambaran

visual, membangun, dan melakukan dokumentasi dalam perangkat lunak (Voutama & Novalia, 2022).

Ada 14 jenis diagram dalam UML yang dapat digunakan dalam proses pengembangan *software*, dimana setiap diagram memiliki fungsi yang berbeda dalam menggambar rancangan. Namun pada penelitian *software development*, diagram yang paling sering digunakan adalah diagram *use case, activity, sequence,* dan *class*.

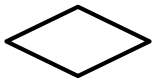
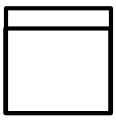
c. *Use Case Diagram*

Simbol	Keterangan
	Sebuah simbol seperti aktor, persona yang berinteraksi dengan sistem.
	*Use case* : abstraksi dan interaksi antara sistem dan aktor.
	Hubungan asosiasi : abstraksi dari penghubung antara *actor* dengan *use case*.
	Hubungan generalisasi : menunjukkan spesialisasi *actor* untuk dapat berpartisipasi dengan *use case*.
<<include >> 	Menunjukkan suatu *use case* merupakan fungsionalitas dari *use case* lainnya.
<<extend>> 	Menunjukkan suatu *use case* merupakan tambahan dari *use case* lainnya jika suatu kondisi terpenuhi.

Gambar 2.2 *Use Case Diagram*
Sumber Gambar: Dicoding (2021)

Diagram yang menggambarkan interaksi pengguna dengan sistem. Selain mudah dipelajari, diagram *use case* juga sangat mudah dimengerti, karena diagram menjelaskan *action* dari pengguna dan sistem itu sendiri, bagaimana sebuah *event* mulai, bagaimana alurnya, dan bagaimana *event* tersebut berakhir. Suatu *event* juga dapat digambarkan oleh lebih dari satu diagram *use case*.

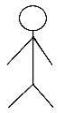
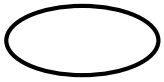
d. *Activity Diagram*

Simbol	Nama	Keterangan
	Status awal	Tanda sebuah status awal dalam proses.
	Aktivitas	Aktivitas dari sistem, biasanya diawali dengan kata kerja.
	Percabangan / Decision	Percabangan dimana ada lebih dari satu pilihan aktivitas.
	Penggabungan / Join	Penggabungan lebih dari satu aktivitas menjadi satu.
	Status Akhir	Status akhir sistem, sebuah diagram aktivitas memiliki sebuah status akhir.
	Swimlane	Pemisah organisasi yang bertanggung jawab terhadap aktivitas yang terjadi.

Gambar 2.3 *Activity Diagram*
Sumber Gambar: Dicoding (2021)

Dapat dikatakan sebuah pengembangan dari diagram *use case*, diagram *activity* menggambarkan proses yang berjalan dalam sebuah sistem yang terdapat juga dalam diagram *use case*, namun digambar dalam bentuk vertikal.

e. *Sequence Diagram*

Simbol	Keterangan
	Sebuah simbol seperti aktor, persona yang berinteraksi dengan sistem.
	*Use case* : abstraksi dan interaksi antara sistem dan aktor.
	Hubungan asosiasi : abstraksi dari penghubung antara *actor* dengan *use case*.
	Hubungan generalisasi : menunjukkan spesialisasi *actor* untuk dapat berpartisipasi dengan *use case*.
<<include >> 	Menunjukkan suatu *use case* merupakan fungsionalitas dari *use case* lainnya.
<<extend>> 	Menunjukkan suatu *use case* merupakan tambahan dari *use case* lainnya jika suatu kondisi terpenuhi.

Gambar 2.4 *Sequence Diagram*
Sumber Gambar: Badoy Studio (2023)

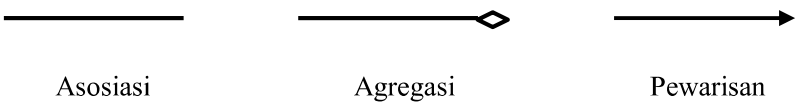
Diagram *sequence* menjelaskan urutan langkah yang dilakukan untuk mencapai sebuah hasil, menjelaskan interaksi antar objek berupa pesan dengan patokan waktu. Digambarkan dalam bentuk garis dengan panah yang menghubungkan satu objek dengan lainnya.

f. *Class Diagram*

Gambar 2.5 *Class Diagram*
Sumber Gambar: Dicoding (2021)

Model UML yang mendefinisikan kelas kelas yang dibuat dalam proses konstruksi sebuah sistem untuk mendeskripisikan strukturnya (Fatmawati, Priandika, & Putra, 2022). *Class diagram* menjelaskan hubungan yang terjadi dalam sebuah sistem, mudah digunakan, cocok untuk model yang sederhana maupun yang kompleks.

Dalam sebuah entitas *class* diagram, terdapat beberapa komponen, atas, tengah, dan bawah. Komponen atas berisi nama *class*, komponen tengah berisi atribut yang mendeskripsikan isi *class* tersebut, dan komponen bawah berisi daftar operasi yang berjalan dalam *class* tersebut.



Gambar 2.6 Hubungan *Class Diagram*
Sumber Gambar: Data Penelitian

Setiap kelas juga memiliki hubungan yang berbeda-beda, tiga jenis hubungan yang terdapat dalam *class* diagram yaitu:

1. Asosiasi

   Hubungan statis yang biasa berarti sebuah *class* memiliki atribut tambahan

2. Agregasi

   Hubungan yang menjelaskan dimana sebuah *class* merupakan bagian dari *class* yang lainnya, namun keduanya dapat berdiri secara individu

3. Pewarisan

   Hubungan dimana sebuah *class* mewarisi semua atribut yang terdapat dalam *class* induknya.

2.3. Objek Penelitian

Solas School of Languages Batam pertama kali dibuka pada tahun 2016 dengan kelas kursus untuk Bahasa Inggris dan Jepang. Guru yang dari awal sudah mengajar disana sudah memiliki kemampuan berbahasa yang kompeten layaknya sebaik *native speaker*, kelas untuk Bahasa Jepangnya pun sudah tersedia mulai dari *level* yang paling dasar hingga *level* yang sudah rumit dan lanjutan. Seiring tahunnya kelas kursus yang tersedia di *Solas* juga bertambah dengan variasinya, mulai dari Bahasa Jerman dan Bahasa Korea, namun pada waktu penelitian dilaksanakan, peneliti mengetahui bahwa kelas Bahasa Korea sudah tidak dilanjutkan lagi. Namun menurut wawancara peneliti dengan Ibu Yunia Wongso selaku komisaris pada *Solas School of Languages*, banyak alumni murid yang belajar di Solas sudah sukses berprestasi dan mendapat pekerjaan di luar negeri, terutama murid-murid yang mengikuti kursus Bahasa Jepang.

2.4. Penelitian Terdahulu

Peneliti memaparkan penelitian, laporan, karya tulis, atau refrensi yang memiliki kaitan dengan penelitian yang sedang dijalankan. Berikut beberapa jurnal yang digunakan dalam pembuatan laporan penelitian ini:

Tabel 2.1 Penelitian Terdahulu

No	Peneliti	Judul & ISSN	Penjelasan
1	Farhan Azharuddin Asyraq, Dhebys Suryani Hormansyah, Mungki Astiningrum	Implementasi FSM (Finite State Machine) Pada Game Surabaya Membara. ISSN: 2614-6371 E-ISSN: 2407-070X Volume 6, Edisi 2, Februari 2020	Membuat sebuah *game* untuk mengenang peristiwa sejarah Indonesia tentang pangeran Diponegoro Perilaku dari pemain dan musuh diatur menggunakan *finite state machine*, sehingga secara otomatis keadaan *player* berbeda ketika musuh sedang jauh, mendekat, berlari.
2	Rian Dwi Susanto, Dodik Arwin Dermawan	Implementasi Finite State Machine dan Algoritma Naïve Bayes pada Game Lord Of Sewandono. ISSN : 2686-2220	Menggunakan metode *finite state machine* untuk membuat *game* tersebut lebih natural dan hidup,. *Finite state machine* pada penelitian ini digunakan

		Volume 03 Nomor 01, 2021	ketika sedang berlari, menyerang, dan bertahan, dimana setiap *action* tersebut akan memiliki animasi yang berbeda.
3	Agustin, Aldino Evel, Susanti, Rahmaddeni	Implementasi Metode Finite State Machine Pada Permainan Tradisional Setatak Berbasis Android. ISSN 2407-4322 E-ISSN 2503-2933 Vol. 8, No. 2, Juni 2021	Menggunakan metode FSM untuk mengatur keadaan pemain terhadap kejadian dalam *game* tersebut. Bagaimana perilaku pemain ketika berhasil, ketika gagal, ketika menang, dan kalah. Penelitian ini berhasil menunjukan bahwa *game* tradisional dapat digunakan untuk menambah pengetahuan terhadap budaya, dan juga disukai oleh anak anak, sehingga secara tidak langsung budaya tradisional tersebut dapat dilestarikan.

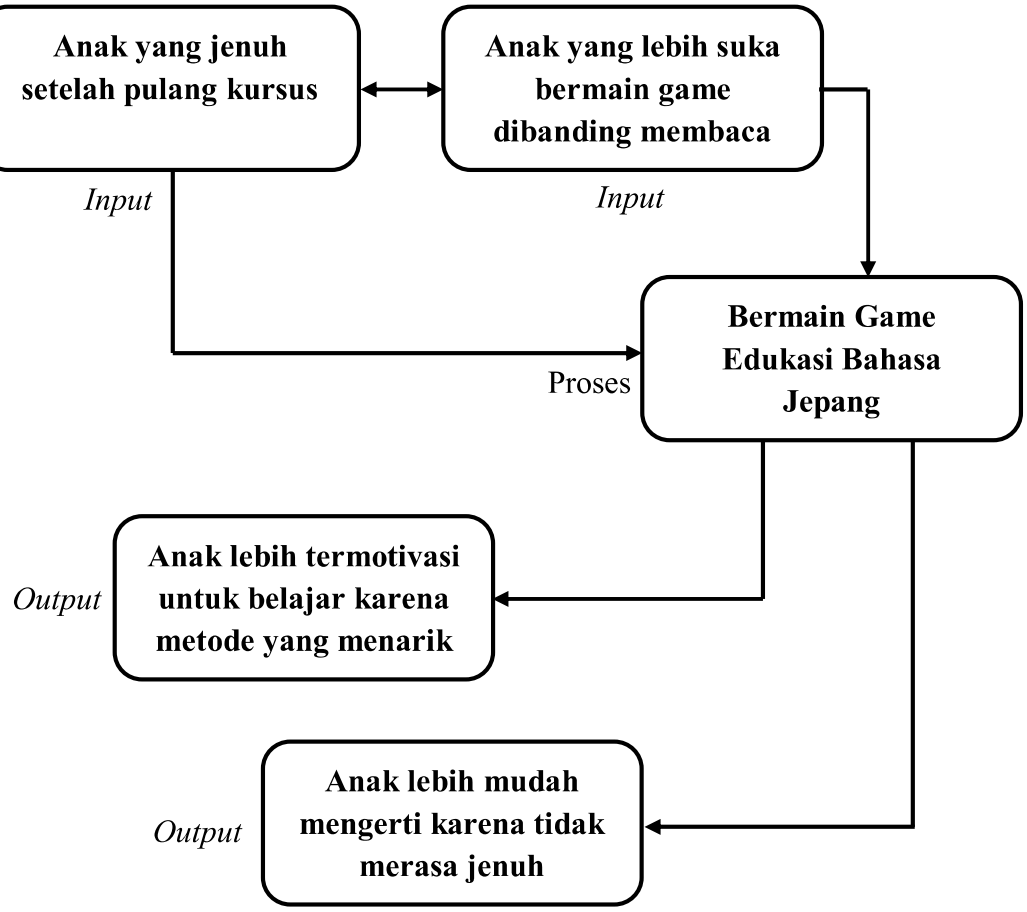
4	Arbansyah	Implementasi Finite State Machine (FSM) Pada Agent Permainan Game Lost Animal At Borneo Berbasis Android. p-ISSN: 2723-567X e-ISSN: 2723-5661 Vol. 3, No. 2, Agustus 2022	Penelitian tersebut menggunakan metode *finite state machine* untuk mengetahui apakah dapat diimplementasikan pada game Lost Anime Borneo. Peneliti menerapkan metode *finite state machine* dalam mendefinisikan perilaku dari pemain, bagaimana state pemain pada awal *game*, pada saat pemain menemukan binatangnya, pada saat pemain belum menemukan binatang ketika waktu habis.
5	Aditya Enggar, Purba Daru Kusuma, Ratna Astuti	Implementasi Finite State Machine Untuk Npc Pada Game 2d Side-Scroll Shooter Implementation Of Finite-State Machine	Menggunakan *finite state machine* untuk mengubah keadaan NPC untuk merespon pemain dalam *game* tersebut, sehingga menjadi lebih seru dan

		For Npc In 2d Side Scroll Shooter. ISSN : 2355-9365 Vol.9, No.3 Juni 2022	menantang. Hal ini juga membuat NPC satu dengan yang lainnya memiliki pola gerakan yang selalu berbeda, alhasil pengalaman bermain lebih dinamis dan tidak membosankan.
6	Muhammad Yoga Altoofa, Titin Fatimah, Dewi Kusumaningsih, Wahyu Pramusinto	Implementasi Finite State Machine Pada Game "Malin Kundang: Simple Platform Game" Dengan Unity Game Engine. ISSN 2962-8628 Volume 2, Nomor 1, April 2023	Peneliti menerapkan sebuah sistem cerdas, yang dimana adalah *finite state machine* dalam merancang musuh dan NPC pada *game* yang dirancang, agar terjadi keadaan yang dinamis secara otomatis, seperti keadaan musuh ketika sedang diam, atau mengejar, keadaan NPC ketika sedang diam, atau berdialog.
7	M. Naufal Azzmi. H1 , Umi Laili	Analyzing The Quality Of Game-Based	Peneliti ingin membuktikan apakah metode ujian atau

	Yuhana, Nawang Sulistyani, Lailatul Husniah	Assessment Design In Basic Arithmetic Operations. Volume 4, Nomor 3, Febuari 2023	*test* dengan menggunakan *game* lebih efektif dari metode lain, khususnya dalam matematika pada penelitian ini. Penulis menggunakan sampel penguji berumur 11 sampai 12 tahun berjumlah 35 anak, dan mendapatkan kesimpulan bahwa 85.7% dari responden setuju menjadikan *game* sebagai metode penguji kemampuan akademis

2.5. Kerangka Pemikiran

Gambaran alur logika yang akan mempermudah jalan penelitian yang sedang dilaksanakan. Berikut kerangka pemikiran pada penelitian "Implementasi Finite State Machine Pada Game Edukasi Bahasa Jepang", peneliti menetapkan anak yang jenuh setelah pulang kursus, dan anak yang lebih senang bermain game dibanding membaca, prosesnya adalah ketika anak anak tersebut ingin istirahat dan bermain game, mereka dapat bermain game edukasi bahasa jepang, yang dimana akan memberi output anak lebih termotivasi karena metode belajar yang menarik, dan materi lebih mudah diterima karena anak tidak merasa jenuh.



Gambar 2.7 Kerangka Pemikiran
Sumber Gambar: Data Penelitian (2024)