# BAB IIIANALISA DAN PERANCANGAN

## Analisa

### Identifikasi Masalah

Pada *Game* atau permainan di era sekarang banyak mengutamakan 3 hal yaitu *Logic*, *Environment*, dan *Sound*. Tujuan penelitian ini yaitu mematahkan *stigma game* ber-*genre* 2 dimensi di dalam industri *Game*. *Environment*, animasi, karakter, *lighting*, dan *Post Process* merupakan tahapan implementasi kedepannya untuk peningkatan kualitas *Game* semakin realistis. Tampilan *user interfaces* dan *particle* akan mendukung *Game* dalam mensimulasikan *Game* yang *fun*. *Story* yang akan dibawakan di dalam *Game* dapat meningkatkan kualitas karakteristik *Game* tersebut dan banyak diminati kalangan pemain baru lainnya. Karena dari *story* atau cerita terdapat klimaks, *problem*, dan *issue* yang nantinya diaplikasikan ke dalam permainan. Dari rangkaian *Game* yang menjadi pembanding akhir nya termuncul-lah ide untuk membuat permainan dengan karakteristik dan warna yang berbeda sehingga para pemain sangat tertarik memainkannya.

### Pemecahan Masalah

Solusi dari berbagai identifikasi masalah yang ada yaitu menggunakan Animasi Karakter 3 dimensi dengan tampilan *environment* 3 dimensi dan terdapat *cinematic camera* yang akan membuat suasana semakin menarik dan kemudian didukung dengan adanya *story* atau cerita disetiap karakter di dalam permaian, yang mana mempunyai tujuan yaitu meningkatkan kualitas permainan. Terdapat *Artiticial Intelligence* yang akan membantu dalam pembuatan dan peningkatan kualitas permainan. Dengan *Artifical Intelligence. logic* permainan akan menjadi realistis karena adanya otomatisasi permainan pada karakter ataupun *environment* dan semua *logic* tersebut akan di-*trigger* oleh pemain. *Engine* yang dipakai tidak lain tidak bukan Unreal Engine. *Engine* ini memiliki ciri dan karakter yang di luar logika manusia, yaitu meningkatkan *futuristic environment* dan membawa dunia nyata menjadi *real*-*time* di dalam *Game* maupun di dalam *Movie*. Banyak perfilm-an memakai *engine* ini untuk dipakai sebagai pembuatan film yang lebih realistis dan mempunyai *engine* tersebut yang dapat menyatukan berbagai elemen dan *asset* menjadi satu kesatuan sehingga tampillah sebuah karya yang akan membuat para penonton merasa tersanjung. Pengunaan *script* yang *art friendly* dan tampilan *environment* 3 dimensi yang menjadi hal utama mengapa para developer pindah ke *engine* ini. Dan dapat disimpulkan Unreal Engine ini merupakan pilihan tepat dalam perancangan penelitian kali ini.

## Perancangan

### Perancangan Data

Berikut merupakan deskripsi darin perancangan sistem yang akan dibuat dalam melakukan desain *game.*

1. Peneliti: Stefanus Salem
2. Judul *Game*: Colleges Life
3. Kategori *Game*: *Game* simulasi
4. Tipe *Game*: *Adventure*
5. *Genre Game*: *Adventure*
6. Pencapaian Permainan: *Reputation*, didapat dari akumulasi antara *Daily Campus* dan *Relationship.*
7. *Daily Campus,* merupakan aktifitas ketika *player* mengikuti perkuliahan dan mendapatkan *score* ketika sudah mengikuti kelas.
8. *Relationship, player* akan mendapatkan poin *relationship* ketika melakukan interaksi pada *character* di dalam permainan. *Player* ditantang untuk mendapatkan *love* yang banyak dari setiap *character* sehingga kemudian mereka akan mendapatkan reputasi yang mana merupakan goal dari permainan ini.
9. Objek Permainan:
	1. *Daily Campus.* Di dalam *user interface daily campus* nantinya terdapat berbagai informasi mengenai jadwal perkuliahan yang *player* akan ikuti, yaitu mengenai waktu kapan perkuliahan akan dimulai. Ketika ingin memulai perkuliahan sebaiknya *player* datang secepat dan setepat mungkin pada waktu kelas dibuka, karena ketika player terlambat maka itu akan mempengaruhi score mereka. Batas *timeout* perkuliahan yaitu 30 menit. Ketika waktu mencapai menit 40 maka kelas akan otomatis tertutup dan *player* akan mendapati bahwa mereka absen kelas dan informasi tersebut dapat dilihat di panel *Daily Campus*, dimana terdapat *icon* kalender silang menandakan *player* telah terlambat atau tidak hadir kelas pada hari itu. *Player* dituntut dapat meningkatkan *status* bar mereka karena *status bar* tersebut berdampak pada *score* yang akan *player* dapati nantinya. Para player juga diharuskan memantau waktu perkuliahan dan menyiapkan status mereka dengan baik.
	2. *Relationship*: merupakan relasi atau hubungan antara *player* dengan karakter di dalam permainan. Setiap harinya mereka hanya memiliki 1 *cache transition love* untuk dapat mereka capai, dan berlanjut kemudian harinya untuk mendapatkan *transition love*. Ketika *player* telah mendapatkan *love* maka merekaakan dihadapi pada *cinematic event* pada tiap *character* yang mereka *trigger.*
	3. Game Timer: Waktu di dalam Permainan yaitu 1 x 24 jam dalam skema waktu yaitu 5 detik *Real Time* per 10 menit di dalam *Game Time*. Batas *timeout* di kampus yaitu jam 07:00 - 18:00, sedangkan batas *timeout* di *PlayGround* 06:00 - 22:00. Disamping itu ketika player berlarut malam maka mereka akan mendapatkan notifikasi untuk dapat keluar dari level atau lokasi tersebut.
	4. 5 *Status Bar*:
		* 1. *Food Status Bar* : dapat ditingkatkan melalui *Mark Food* yang disediakan di dalam *Game*.
			2. *Sleep Status Bar* : dapat ditingkatkan melalui *Mark Sleep* yang disediakan di dalam *Game*.
			3. *Study Status Bar* : dapat ditingkatkan melalui *Mark Study* yang disediakan di dalam *Game*.
			4. *Fun Status Bar* : dapat ditingkatkan melalui *Mark Fun* yang disediakan di dalam *Game*.
			5. *Clean Status Bar* : dapat ditingkatkan melalui *Mark Clean* yang disediakan di dalam *Game*.



Gambar 3.1 Gambar *Status Bar*

* 1. *Mood*: *Mood* adalah konversi rata-rata dari kelima *Status Bar* yang telah disediakan oleh *Game*. Ketika *Mood* mencapai berada pada posisi status 70% ke atas maka statusnya yaitu *Happy.* Dan ketika player mengalami status *Very Sick* maka pemain akan mendapatkan notif sakit dan akan melewati 2 Hari setelah hari itu atau akan langsung menghadapi *Graduation Day* ketika mencapai hari ke 29.



Gambar 3.2 Gambar *Mood*

1. Elemen dalam *Game*:
2. Teknologi: Unreal Engine, Blender, Adobe XD, Adobe Photoshop, dan Adobe Premiere Pro
3. *Level*:
4. Rumah: Tempat dimana nantinya para pemain dapat meningkatkan *Status Bar* mereka. Di dalam Rumah *Player* dapat meningkatkan *Status Food*, *Fun*, *Clean*, *Sleep*, dan *Study*. Diharapkan *player* nantinya mereka dapat mempersiapkan status mereka agar setelah perkuliahan selesai mereka mendapat *score* yang baik. Assets yang dibuat bisa didapat dari Epic Marketplace di situs berikut ini.
https://www.unrealengine.com/marketplace/en-US/product/cartoon-house-interiors



Gambar 3.3 Sketsa Level di Rumah

1. Kampus: Suasana dimana *player* dapat melakukan interaksi dengan *Non*-*Playable Character* dan meningkatkan relasi mereka dengan *Non*-*Playable Character* tersebut. Terdapat 3 kelas yang dapat player nanti masuki yaitu kelas A, B, dan C. Para pemain juga bisa meningkatkan *Status Bar Food* dan *Clean* ketika berada pada *Marker* yang ada di Kampus terutama di Kantin dan Toilet untuk meningkatkan *Status* mereka.
https://www.unrealengine.com/marketplace/en-US/product/olbert-s-low-poly-school



Gambar 3.4 Gambar Sketsa di Kampus.

1. *PlayGround*: Para pemain dapat melakukan interaksi dengan *Non*-*Playable Character. Player* bisa menikmati pemandangan yang indah dan eksplorasi ketika berada di *level playground.*
https://www.unrealengine.com/marketplace/en-US/product/stylized-landscape



Gambar 3.5 Gambar Sketsa di Taman Bermain

1. *Sound / music*: di dalam rumah terdapat musik sendu, sedih, dan penuh harapan. Ketika berada di kampus terdapat musik bersuasana semangat, enjoy, dan penuh motivasi. Namun ketika berada di PlayGround terdapat musik senang, semangat, tenang dan penuh harapan.
https://www.epidemicsound.com
2. *Graphic*:
3. *Actor*: *Main Character* dan *Non*-*Playable Character* yang dipakai di dalam *game* adalah sebagai berikut.
https://www.unrealengine.com/marketplace/en-US/product/polygon-city-characters



Gambar 3.6 Gambar Seluruh Karakter dalam *Game*

1. *Non Character*: Kampus *Mesh*, *House Mesh*, dan *PlayGround Mesh*.



Gambar 3.7 Gambar seluruh *Mesh* di dalam *Game*

1. *Game Story* :
2. Caroline *Event*. Caroline adalah salah satu karakter pertama yang ditemui oleh *Player* ketika berada di *opening scene*. Ketika permainan berlangung karakter ini dapat ditemui di *PlayGround* untuk dapat berinteraksi dengan *Player*.
3. John *Event.* John memiliki sifat kebapak-an dan terdapat cerita dimana John memberitahu hal-hal mana yang benar dan salah yang harus *player* ikuti ketika berada di kampus. *Event* tersebut dapat *player trigger* pada karakter ini ketika berada di kampus tepatnya di dalam kantor kampus.
4. Jackson *Event.* Jackson adalah seorang buruh pekerja yang menyempatkan waktunya untuk pergi ke *PlayGround* untuk menikmati suasana pemandangan yang indah ketika di *PlayGround*. Tentunya *player* dapat men-*trigger* karakter ini ketika berada di *PlayGround*.
5. Jeanne *Event.* Jeanne adalah seorang paramedis di kampus. Karakteristik Jeanne adalah seorang yang ramah, suka memberi, suka menolong, dan dia adalah seorang yang selalu siap siaga. Tentunya karakter ini dapat *player trigger* ketika berada di kampus. Dan karakter ini memiliki *Artificial Intelligence* yaitu melakukan otomatisasi mondar-mandir di sekitar kampus.
6. Christ *Event.* Christ adalah seorang turis dari Belgium, yang mana Christ adalah seorang yang sangat suka dengan hal-hal legendaris. Dia menyempatkan waktunya berada di *PlayGround* hanya untuk meneliti dan melihat barang-barang peninggalan di lokasi tersebut. Tentunya *player* dapat melakukan *trigger* pada karakter tersebut ketika berada di *PlayGround*.
7. Unsur *Fun* atau *Game play* : Para *player* mempunyai rasa tergesa-gesa ketika mau mengikuti kelas karena mereka harus mempersiapkan *status* mereka dengan baik.

### Perancangan *User Interface* / *Mock*-*up* aplikasi

Berikut adalah tampilan *Main User Interfaces*. Disinilah *Player* pada umumnya berinteraksi ketika berada dalam permainan. *Player* berlari menggunakan *Joystick*, berlari, Jalan, dan melompat sesuai keinginan *Player* saat berada dalam permainan. Tersedia *Configuration Button* yang nantinya menuju ke tampilan *Game Setting Panel.* Tersedia juga 5 *Status Bar* yang sudah di *setup*, baik itu *Food*, *Clean*, *Fun*, *Sleep*, dan *Study Panel Bar* di sebelah kiri atas. Disertai *Mood Panel Status* disebelah kiri *Status Bar.*
https://www.flaticon.com/

****

Gambar 3.8 Gambar *Main User Interface*

Berikut adalah tampilan *Mock*-*Up* untuk *Game Settings Interface*. Berikut terdapat *panel Game Settings* untuk meng-konfigurasi *Graphic Quality*, *Frame Rate*, dan *Master Volume*. Tersedia *Save* and *Load* dan *Guides Button* untuk mengarah ke *Save* and *Load Panel* dan *Guides Panel.*

****

Gambar 3.9 Gambar *Game Settings Panel User Interface*.

Interface berikut bertujuan untuk menginformasikan pemain untuk mengetahui tingkat reputasi.

****

Gambar 3.10 *Relationship Panel User Interface*

Berikut adalah tampilan *Daily Presence Panel Interface*. Memili tujuan yaitu lamanya waktu permainan berakhir. Dalam tiap *Panel* nya terdapat informasi Hari, Kelas, Waktu Kelas dimulai, Absensi, dan *Score*. Terdapat Tombol *Close* dan *Next* atau *Previous Button* untuk menavigasikan *Panel Interface*.

****

Gambar 3.11 Gambar *Daily Presence Panel User Interface*

Berikut adalah tampilan *Guides Interface*. Bertujuan untuk menginformasikan, mengontrol, dan mengkonfigurasikan permainan. *Panel* ini muncul ketika *Opening Scene* atau ketika *Player* menekan *Guides Button* yang ada di *Game Settings* dan memilih berbagai jenis *Guides* di dalam permainan.



Gambar 3.12 Gambar *Guides User Interface*

Berikut adalah tampilan *Quest Panel Interface*. *Panel* ini hanya tampil ketika saat *Opening Scene* berlangsung, dan mengarahkan pemain untuk melakukan suatu hal.



Gambar 3.13 Gambar *Tips User Interface*

Dalam *Panel* ini Pemain diarahkan untuk memilih *Gender* atau *Skin*, dan dapat melihat tampilan *character,* di tambah navigasi state yaitu *walking*, *running*, dan *sprinting* di sebelah kiri dari navigasi karakter. Pemilihan *Gender* terdapat 2 yaitu Perempuan dan Laki-laki kemudian Varian *Skin* terdapat 4.



Gambar 3.14 Gambar *New Game User Interface*

Berikut adalah tampilan *End Game Score*. *Panel* ini berfungsi untuk menampilkan *Score* dari *Player Reputation* yang telah diraih. Berikut terdapat informasi *Gender* dan *Name* serta Persentasi *Reputation* yang telah diraih oleh pemain.



Gambar 3.15 Gambar *End Game Score*

### Perancangan Sistem

Pada penelitian ini nantinya sistem perancangan yang dipakai yaitu *Finite State Machine* (FSM). Baik itu *Logic Player Controller*, *User Interfaces*, *Flow Story*, *Artifical Intelligence*, dan *Functionality* lainnya sangat membantu kebutuhan perancangan permainan. *Finite State Machine* ini banyak dipakai untuk *Logic* pembuatan *Game*. Tentunya *Game Simulator* sangat dibantu dengan adanya perancangan sistem ini.

*State* awal adalah *Main User Interfaces* tampilan dimana permainan dimulai dan *state interface panel* inilah yang akan tampil. Ketika pemain menekan tombol “*Relationship”* maka akan tampil “*Panel Relationship”*, namun ketika pemain menekan tombol “*Close Relationship”* maka akan balik ke *Main User Interfaces*. Ketika menekan tombol *Daily Campus* maka akan tampil *Daily Campus Panel*, sebaliknya jika tombol *Close* dari *Campus Panel* di tekan maka akan balik ke tampilan *Main User Interfaces*. Ketika ingin membuka “*Game Settings”* maka *Player* menekan tombol *Game Settings*, dan sebaliknya ketika menekan tombol *Close* pada *Game Settings Panel*. Dan ketika berada di dalam atau di *State Game Settings Interfaces* maka *Player* dapat memilih kembali baik itu ke tampilan *Guides Interfaces* atau ke *Save* and *Load Interfaces* dengan menekan tombol *Game Settings* atau tombol *Save* and *Load*.



Gambar 3.16 Gambar *Main User Interfaces State Machine*

Berikut adalah tampilan *Player Controller State Machine* ketika berada dalam posisi *State Game Play* dalam *Main User Interfaces*. *Logic* berikut menunjukkan *State* ketika *Controller* di *trigger* oleh pemain menggunakan *JoyStick* dan tombol lainnya untuk merubah *State* berikut. *State* awal adalah *Standing* kemudian ketika *Joystick* di gerakkan maka *State*-nya adalah *Running*, ketika dilepas maka akan *Standing*. Terdapat *State Jumping* dimana tombol ini dapat di-*trigger* kapan pun selama itu tidak termasuk bagian dari *state Crouching* dan *Walk Crouching* sebelumnya. Terdapat *State Walking* dimana dapat di *Trigger State* ini ketika *Standing* ataupun ketika berada dalam *State* lainnya yang mana *Backward State* nya adalah *Standing*. *State Jumping* dapat di-*trigger* ketika tombol *Jump* ditekan dan ketika *state* nya adalah *Standing* ataupun *State* lainnya selama *Backward State*-nya adalah *Standing*. *Crouching idle* ketika tombol *Crouch* ditekan dan ketika sambil mengarahkan *Joystick* maka akan menjadi *State Crouching Walk*. Ketika *Running* dan tombol *Sprinting* ditekan maka akan menjadi *State Sprinting*, sebaliknya ketika di lepas akan kembali ke *State Running*. Ketika *State* ini *active* maka ketika *Joystick* digerakkan maka terjadi *State Walking Injured*, dalam satu waktu tombol *Jump* ditekan maka menjadi *State Jumping Injured*, sebaliknya ketika *State Functionality* sudah tidak berfungsi atau *active* maka akan kembali ke *Origin State* semisal *Player Injured* atau *Standing*.



Gambar 3.17 Gambar *Player Controller State Machine*

Berikut adalah tampilan *Status Bar State Machine*. *State Machine* ini berfungsi untuk menunjukkan *Logic* dari *Status Bar* baik itu *Increase Status Decrease Status* maupun *Update Mood*. *Idle State Status* adalah awal mula dimana *State* ini berlangsung ketika *Status Bar* tidak ter-*trigger* oleh *Functionality Increase* ataupun *Decrease Status* maka disebut *Idle State Status*. Ketika dalam posisi *Restriction Status In Place* yaitu *Mode* untuk *Handle Decreasing Status m*aka akan terjadi *Decreasing Status State*. Di dalam *State* tersebut terdapat Animasi untuk memanipulasi *Status Bar* sehingga menginformasikan *Player* bahwa *Status Bar* sedang menurun. Sebaliknya Ketika melakukan sesuatu atau *Trigger Increase Status*, maka *State* akan berpindah dari *Idle State* ke *Increasing Status State*. *State* ini terdapat animasi untuk menunjukkan kepada *Player* bahwa *Status Bar* sedang naik. *Trigger Increase Status* dapat dicapai melalui *Trigger Status Behaviour* seperti *Food*, *Fun*, *Clean*, *Study*, dan *Sleep*. Ketika *Increase Status* atau *Animate Status* dan *Increasing Bar* selesai maka akan kembali ke *Idle State Status*. Selanjutnya ketika *Finish Trigger Status* artinya ketika *Status* apapun telah selesai dianimasikan kemudian akan *update Status Bar* dan akan kembali ke *State Update Mood* untuk memperbaharui *Status Mood* dan memunculkan *Mood Icon* ketika *State Mood* berubah.



Gambar 3.18 Gambar *Status Bar State Machine*

Berikut adalah tampilan *Mood State Machine*. *State Machine* ini berfungsi untuk Menampilkan *State Mood* dalam bentuk *Icon* di dalam *User Interfaces* di kiri atas layar *Interfaces*. Persentase tersebut dapat dicapai melalui akumulasi dari ke lima *Status Bar*, konsep yang dipakai yaitu rata-rata dari ke lima *Status Bar* tersebut kemudian didapati persentase *Status Average.* Kemudian ketika posisi berada pada *Very Sick State* maka *Player* akan mendapati notifikasi Sakit dan akan melewati hari selama 2 hari. Perpindahan dari tiap *State* terdapat Transisi *Icon* dan terdapat 3 varian *Icon* untuk menunjukkan *Player Mood Status* baik itu *Normal*, *Sad*, ataupun *Sick*.



Gambar 3.19 Gambar *Mood Finite State Machine*

Berikut adalah tampilan *Time State Machine*. *Behaviour* ini berfungsi untuk menunjukkan *Logic* kalkulasi dari perubahan Waktu di dalam permainan. *State* awal akan berada dalam *Calculate Time to Delay*, artinya terdapat *State Init Timer* untuk men-*simulate Timer* berdasarkan *default setting* yang sudah di *set up*. Ketika menit dalam satuan waktu permainan ter-*trigger* maka *Update Minute of Time State* akan jalan. *Update* tersebut berupa pembaharuan dari *Time Interface Panel* disebelah kanan atas di dalam *Timer Panel*. Ketika *Update Minute* selesai maka akan balik ke *State* awal yaitu *Calculate Time* kembali. Kemudian apabila *Hour of Time* ter-*trigger*, artinya jam dalam satuan waktu permainan ter-*trigger*. Maka akan *Update Hour of Time* yang terdapat dalam *User Interfaces Time Panel* di sebelah kanan atas, begitu perbaharui *Hour of Time Variable* dalam *Set up Timer* di dalam permainan. Kemudian apabila *Update Hour* selesai maka akan ada *delegate Finish Update Hour* dimana *State* ini untuk melakukan *Check Weather* pada *Timer Interfaces Panel* untuk mentransisi *Weather Icon*. Varian *Icon* terdapat 3 yaitu *Morning*, *Sunny* dan *Evening Icon* tergantung dari *Weather State* yang telah di *set up* sehingga adanya transisi disana.



Gambar 3.20 Gambar *Time State Machine*

Berikut adalah tampilan *Weather State Machine*. *State Machine* ini berfungsi untuk menunjukkan *Logic* ketika *State Weather* bertransisi. *State a*wal dari *State Machine* ini adalah *Morning State*. Ketika *Hour* mencapai angka 10 keatas maka *Sunny State* akan berjalan. Apabila *Hour* mencapai 5 keatas maka *Morning State* akan *active*. *Update Interfaces State* ini berfungsi untuk memperbahuri *State Interfaces* dari *Weather Icon* itu yang terdapat di sebelah kanan atas *Panel User Interfaces* di dalam *Timer Interfaces Panel*. Total terdapat 3 varian *Weather Icon* untuk mensimulasikan kondisi *Weather Icon* saat itu sesuai satuan waktu *Hour* yang ada di dalam permainan.



Gambar 3.21 Gambar *Weather State Machine*

Berikut adalah tampilan *Relationship State Machine*. *State* awal ketika permainan dimulai yaitu *Intro State Dialogue*. Ketika *Finish* maka lanjut ke *Loop End Intro State Dialogue* yaitu *State* ketika *State Dialogue* pada hari itu telah tercapai dan akan terjadi *Loop* selama belum ter-*cache*. Dan apabila melakukan *interact* kembali ke *Non*-*Playable Character* maka *Loop End Intro State Dialogue* akan kembali *active*. Apabila *Cache relationship* ter-*trigger* setelah *Intro State Dialogue* selesai maka akan lanjut ke *State Normal Dialogue State*. Ketika selesai *Normal Dialogue State* atau telah *Interact* dengan *Non-Playable Character* dalam *State Normal Dialogue* maka akan lanjut ke *Loop End Normal Dialogue State*. *State* ini akan terjadi *Loop Dialogue* apabila *Normal Dialogue* telah dilewati, dan *State* ini akan kembali *active* dan berjalan ketika melakukan *Interact* kembali ke *Non*-*Playable Character* tersebut. Kemudian ketika *Cache Relationship* ter-*trigger* dan *transition Love* tidak sama dengan 3 maka akan kembali ke *Normal Dialogue State*. *Transition Love* menunjukkan *clamp max* dari *transition* yang sudah di *Set up* yaitu berjumlah 3 *transition*. *Transition* tersebut dapat ditingkatkan melalui *Trigger Normal Dialogue State*. Apabila *transition Love* sudah mencapai 3 dan ter-*cache* maka akan pindah *State* ke *Love Dialogue State*. *Love Dialogue State* merupakan *Cinematic State*, dimana terdapat *Cinematic Scene* dari tiap *Non*-*Playable Character* permainan yang berbeda-beda dan mempunyai konteks dan ciri khas disetiap ceritanya. Kemudian ketika *Love Dialogue State* selesai maka akan terjadi *Loop Love Dialogue State* kembali, yang mana akan *Loop* ketika di *interact* kembali pada *Non-Playable Character* tersebut dan belum ter-*cache* pada hari itu. Ketika Cache Relationship true dan love belum mencapai 3 maka akan kembali ke *State Normal Dialogue State*. *Love* di dapat ketika *Transition Love* telah mencapai 3 dan ter-*cache Relationship*. Dan akan terjadi *loop* atau perulanganterus ketika *State* ini ter-*trigger*.



Gambar 3.22 Gambar *Relationship State Machine*

Berikut adalah tampilan *Daily Presence State Machine*. *State Machine* ini menunjukkan *Logic* dari *Daily Presence* atau ketika *player* hadir perkuliahan setiap harinya. *State* awal *Delegate Hour to Check the Attendance Class*, *State* ini bertujuan untuk *check* apakah kelas akan dibuka atau tidak. Kemudian ketika *delegate* tersebut ter-*trigger* maka akan lanjut ke *State Open the Class*. *Open the Class* ini mensimulasikan *particle* berupa *Mark* di depan kelas yang menunjukkan bahwa kelas telah dibuka. Kemudian terjadi fungsi *delegate* baik itu *Player* men-*trigger Class* dan akan lanjut ke *Enter the Class and Play Cinematics State* atau setelah 30 menit maka akan ter-*trigger Close the Class* atau *Particle Mark* akan ter-*close* dan *Player* tidak dapat menghadiri kelas pada hari itu dan otomatis *Score* akan mendapatkan D.



Gambar 3.23 Gambar *Daily Presence State Machine*

Berikut adalah tampilan *Game State Behaviour State Machine*. *State* awal yaitu *Start Game*. Ketika *Scene Start Game* telah selesai maka akan lanjut ke tahap selanjutnya yaitu *Main Menu Interfaces.* Pada *State New Game Player* akan dikhususkan untuk membuat karakter, baik itu *Gender* maupun *Skin*. Ketika *player* sudah sangat yakin dengan pilihannya, selanjutnya masuk ke tahap *State Opening Scene*. Di *State* ini terdapat 3 Fase yaitu fase *Opening Scene Playground*, *House*, dan *Campus*. Di dalam *Scene* terdapat *Cinematic Event* dan *Dialog* antar *Character.* *Player Name* akan terpakai saat *Player* melakukan dialog pad *Non*-*Playable Character* dan nama *Player* akan disebutkan di dalam dialog tersebut. Setelah *Opening Scene* selesai kemudian terdapat *Guides* dan *Quest* di sela *scene*. Pada *State Play Game* ini maka permainan akan dibebaskan kepada *Player* dan sesuai dari karakter si pemain dalam menjalani permainan nya ketika pemain bermalas-malasan akan mendapat *Score* Besar, Ketika Rajin akan mendapatkan *Score* Bagus. Terdapat *Cinematic* pada state ini dan diakhiri dengan animasi *Scoring* dari *Player*. *State* selanjutnya apabila *Player* menekan tombol *Load Game* pada *Main Menu Interfaces* maka *Player* akan melakukan *Loading* semua Variabel pada terakhir kali *Player* telah Save permainan.



Gambar 3.24 Gambar *Game State Behaviour State Machine*

## Rancangan Pengujian

Rancangan Pengujian yang akan dipakai yaitu Rancangan *Black Box*. Ketika ada *Bug Player* akan menginformasikan Developer untuk melakukan perbaikan dan memperbaharui permainan sampai versi yang tidak ditentukan. Terus akan dilakukan *improvement* sehingga permainan yang dibuat semakin rapih dan clear dari *Bug*. Dengan ada nya *improvement* peminat *Player* lain akan bertambah seiring berjalannya waktu dan rating permainan akan semakin meningkat.

Tahapan melakukan blackbox pertama kali yaitu melakukan *Build Game,* sehingga dapat dimainkan oleh *Client* atau *Player*. Setelah di *build* maka akan terdapat proses *Upload* ke *platform PlayStore,* Kemudian dilengkapi dengan deskripsi dan informasi pembaharuan terbaru. Seteleah melakuan pembagian *link* maka disertai juga dengan pemberitahuan kepada para *tester* untuk mengisi *feedback* setelah melakukan permainan. Setelah mendapatkan *List Feed back* atau *Bug*, kemudian tahap selanjutnya yaitu memperbaiki *Bug* tersebut. Perulangan *flow* akan tetap dilakukan sampai *Feedback Bug* tidak ada.



Gambar 3.25 Gambar Rancangan *Black Box*