BAB IV
ANALISIS KERJA PRAKTEK

IV.1 Input

Dalam proses pembuatan *game* simulasi pendaftaran mahasiswa di Universitas Bale Bandung ada beberapa program yang harus di buat berdasarkan hasil diskusi dengan *game designer* antara lain:

1. *Movement Player*
2. *Change Camera Position*
3. *Dialogue Player*
4. *Open Door*
5. *Open Level*
6. *Energy System :*
    1) *Hunger System*
    2) *Stamina System*
7. *Inventory System*
8. *Player Objective (Quest)*
9. *Player Running*
10. *Main Menu*

IV.2 Proses

Setelah melakukan pengenalan lingkungan kerja pada awal pelaksanaan kerja praktek, selanjutnya proses kerja praktek dapat dibagi menjadi beberapa tahap, yaitu eksplorasi, pembangunan perangkat lunak, dan pelaporan hasil kerja praktek.

IV.2.1 Eksplorasi

Tahap eksplorasi dimulai dengan melakukan eksplorasi mengenai *engine* yang cocok untuk aplikasi *game* ini. Untuk *engine* penyusun kerja prakterk membandingkan antara *UNITY* dari Unity Technologies dan *Unreal Engine 4* dari Epic Games. Perbandingannya meliputi *system requirements*, bahasa yang di pakai masing masing *engine game tersebut,* dan kemudahan dalam pembuatannya (*Assembly*). Dari hasil perbandingan tersebut yang

digunakan adalah *Unreal Engine 4* karena lebih mudah dalam segi pembuatannya.

Untuk mendukung pelaksanaan pembuatan *game*, diperlukan pula pengetahuan mengenai bahasa yang digunakan. Dengan demikian, pemrograman terhadap pembuatan *game* dapat berjalan dengan lancar. Dan untuk menyesuaikan program sesuai pendaftaran maka perlu juga melakukan eksplorasi bagaimana proses pendaftaran itu berlangsung.

Proses eksplorasi masih berlangsung selama pembangunan aplikasi *game* ini. Hal ini dimaksudkan untuk menyelaraskan antara hasil eksplorasi dengan penerapannya pada aplikasi *game* yang sedang dibangun.

IV.2.2 Pembangunan Aplikasi Game

Pembangunan aplikasi *game* ini dimulai dengan analasis desain hasil diskusi dengan desainer. Selanjutnya berdasarkan desain tersebut mulailah program-program dibuat. Untuk memastikan aplikasi *game* yang dihasilkan sesuai dengan desain dan berfungsi dengan semestinya, dilakukan beberapa kegiatan pendukung seperti *testing*, *bug fixing*, dan optimasi performansi, agar program sesuai dengan desain *testing* dilakukan lebih dari satu kali.

Dengan pendekatan *prototyping* pada tahap eksplorasi, pengembangan aplikasi *game* ini membutuhkan waktu yang banyak. Untuk memudahkan digunakan pada komputer yang berbeda, dibuatlah *executable*(*.*exe*).

IV.2.2.1 Program

Dalam program pada aplikasi *game* pendaftaran mahasiswa ada beberapa *input* yang digunakan dari *hardware* seperti *keyboard* dan *mouse*. *Input* ini dibagi menjadi 2 ada *Axis Mappings* dan *Action Mappings*, apa itu *axis mappings* dan *action mappings* berikut adalah penjelasannya *Action and Axis Mappings provide a mechanism to conveniently map keys and axes to input behaviors by inserting a layer of indirection between the input behavior and the keys that invoke it. Action Mappings are for key presses and releases, while Axis Mappings allow for inputs that have a continuous range*.

Berikut ini adalah *input-input* yang digunakan pada aplikasi *game* ini :

Action Mappings	
Name	***Key***
Jump	Space
Run	Left Shift
Discard Item	Left CTRL
Open Door Keyboard	G
Interact	E

Axis Mappings			
Name	***Key***	***Scale***	**Keterangan**
Move Forward	W	1	Atas
	S	-1	Bawah
	Up (↑)	1	Atas
	Down (↓)	-1	Bawah
Move Right	D	1	Kanan
	A	-1	Kiri
Turn Rate	Right (→)	1	Kamera Ke Kanan
	Left (←)	-1	Kamera Ke Kiri
Turn	Mouse-X	1	Kamera Ke Kiri - Kanan
Look Up	Mouse-Y	-1	Kamera Ke Atas - Bawah

Table IV.1 Tabel *Controller*

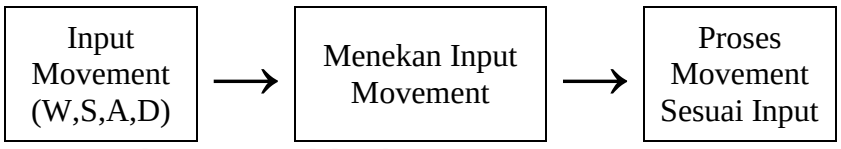
Untuk *scale* disini berperan sebagai pengali atau *multiplier* untuk mengakumulasikan nilai *axis*. Berikut ini adalah penjabaran dari program-program yang di kembangkan :

IV.2.2.1.1 *Movement Player*

Program ini dibuat agar *player* bisa bergerak, gerakan *player* di program ini adalah 8-dir (*direction*/arah) antara lain maju, mundur, diagonal, kanan, kiri. Ada 2 *axis mappings* yang dipakai yaitu MoveForward dan MoveRight. Berikut adalah penjelasan dan model teknis dari program ini :

- Maju
  - Untuk bergerak ke depan *user* menekan tombol di keyboard W atau Up (↑) secara lama (*long press*).
  - Selanjutnya *player* akan bergerak ke depan.

- Mundur
  - Untuk mundur ke belakang *user* menekan tombol di keyboard S atau Down (↓) secara lama (*long press*).
  - Selanjutnya *player* akan mundur ke belakang.
- Kanan
  - Untuk bergerak ke kanan *user* menekan tombol di keyboard D secara lama (*long press*).
  - Selanjutnya *player* akan bergerak ke kanan.
- Kiri
  - Untuk bergerak ke kiri *user* menekan tombol di keyboard A secara lama (*long press*).
  - Selanjutnya *player* akan bergerak ke kiri.
- Diagonal
  - Untuk bergerak secara diagonal *user menekan* tombol di keyboard W dan D atau W dan A atau S dan D atau S dan A secara lama (*long press*).
  - Selanjutnya *player* akan bergerak diagonal sesuai tombol yang di tekan, seperti :
    - W dan D : bergerak ke kanan atas.
    - W dan A : bergerak ke kiri atas.
    - S dan D : bergerak ke kanan bawah.
    - S dan A : bergerak ke kiri bawah.



Gambar IV.1 Model Teknis Pemograman Movement

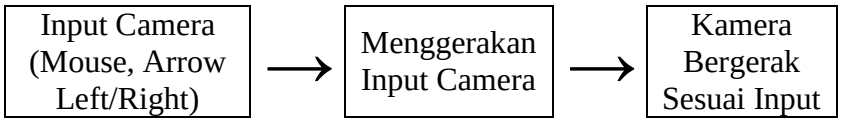
IV.2.2.1.2 *Change Camera Position*

Program ini dibuat agar pada saat player berjalan bisa dengan kamera, misal pada saat maju kedepan dan ingin belok ke kanan bisa dengan hanya menggerakanna kamera ke kanan.

Ada 3 Axis Mappings yang dipakai yaitu TurnRate, Turn dan LookUp, Berikut adalah penjelasan dan model teknis dari program ini :

Untuk TurnRate : Right (→), Left (←) dan Turn : mouse-x dan LookUp : mouse-y :

- Right (→)
  - Jika *user* menekan secara lama *(long press)* pada tombol di keyboard Right (→).
  - Maka kamera akan bergerak memutar ke kanan (berlawanan arah jarum jam) dan *player* sebagai pusatnya.
- Left (←)
  - Jika *user* menekan secara lama *(long press)* pada tombol di keyboard Left (←).
  - Maka kamera akan bergerak memutar ke kiri (searah jarum jam) dan *player* sebagai pusatnya.
- Mouse-X
  - Jika *user* menggerakan mouse ke kanan atau ke kiri.
  - Maka kamera akan bergerak sesuai arah yang dituju (hanya kiri dan kanan) dan *player* sebagai pusatnya.
- Mouse-Y
  - Jika *user* menggerakan mouse ke atas atau ke bawah.
  - Maka kamera akan bergerak sesuai arah yang di tuju (hanya atas dan bawah) dan *player* sebagai pusatnya.



Gambar IV.2 Model Teknis Pemograman Kamera

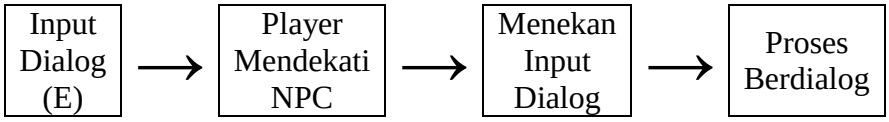
IV.2.2.1.3 *Dialogue Player*

Program ini dibuat untuk bisa berdialog dengan *non player character* (NPC). Kenapa perlu ada dialog, karena pada *game*

simulasi ini *player* diharuskan berdialog dengan beberapa orang karena pada tahap eksplorasi orang yang daftar harus berdialog dengan beberapa orang maka dari itu program ini dibuat.

Program ini memakai *action mappings* berupa Interact : E, Berikut adalah penjelasan dan model teknis dari program ini :

- Untuk bisa berdialog user harus menggerakankan *player* ke npc terdekat.
- Maka akan muncul tulisan "E To Interact".
- Disini *user* harus menekan tombol di keyboard yaitu E.
- Maka *player* akan berdialog dengan npc tersebut.



Gambar IV.3 Model Teknis Pemograman Dialog

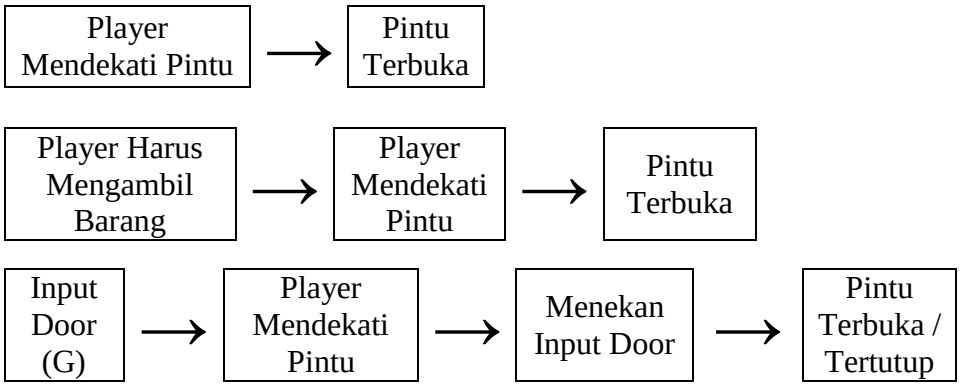
IV.2.2.1.4 *Open Door*

Program ini dibuat supaya *game* terlihat sama dengan dunia nyata yang bisa membuka pintu. Program ini memakai *action mappings* berupa OpenDoorKeyboard : G, Berikut adalah penjelasan dan model teknis dari program ini :

- Untuk program ini *user* harus menggerakan *player* ke pintu yang bisa di buka.
- Maka akan muncul tulisan "G" dan dibawah nya ada tulisan "Open".
- Disini *user* harus menekan tombol di keyboard yaitu G.
- Maka pintu akan terbuka dan tulisan akan berubah menjadi "G" dan dibawahnya ada tulisan "close".
- *User* bisa mencoba menekan kembali tombol "G" dan pintu akan tertutup.

Program ini, juga ada sebelum pintu terbuka *user* harus mengambil suatu barang misal nya kertas, jika belum di ambil

maka pintu tidak akan terbuka, tapi jika sudah di ambil maka pintu akan terbuka secara otomatis.



Gambar IV.4 Model Teknis Pemograman Open Door

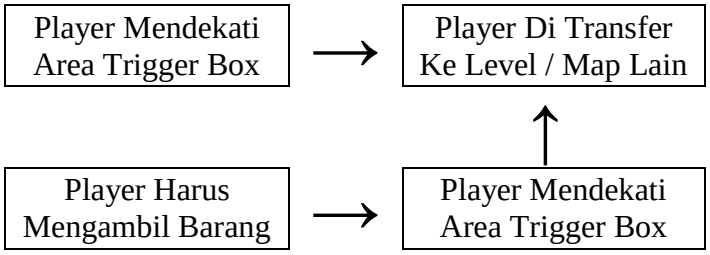
IV.2.2.1.5 *Open Level*

Program ini dibuat karena pada game ada beberapa tempat yang harus ditukunjungi, dan dikarenakan ini tidak sepenuhnya *open world.* Apa itu *open world*, *open world* merupakan sebuah genre dalam *game*, dimana *player* bisa menjelajahi dan mendekati tujuan dengan bebas.

Pada program ini tidak memakai suatu *key* atau *user* tidak harus menekan sesuatu dalam keyboard. Berikut adalah penjelasan dan model teknis dari program ini :

- Pertama user harus menggerakan *player* ke sebuah *triggerbox.* (*triggerbox* disini sebagai pemicu akan terjadinya event jika dan hanya jika *player* ada di dalam *trigerbox* tersebut).
- Jika sudah berada dalam *triggerbox* tersebut.
- Maka player akan di transfer ke level selanjutnya atau ke level yang dituju.

Program ini juga bisa memakai sistem seperti pada buka pintu yaitu harus mengambil barang terlebih dahulu.



Gambar IV.5 Model Teknis Pemograman Open Level

IV.2.2.1.6 *Energy System*

Program ini dibuat agar *player* memang mempunyai apa yang manusia rasakan seperti kelaparan atau stamina berkurang. Di program ini *player* diharuskan memakan atau mengambil barang yang ada di jalanan sebagai *energy*. Disini ada 2 program yaitu *Hunger System* dan *Stamina System*. Berikut penjelasan masing masing

IV.2.2.1.6.1 *Hunger System*

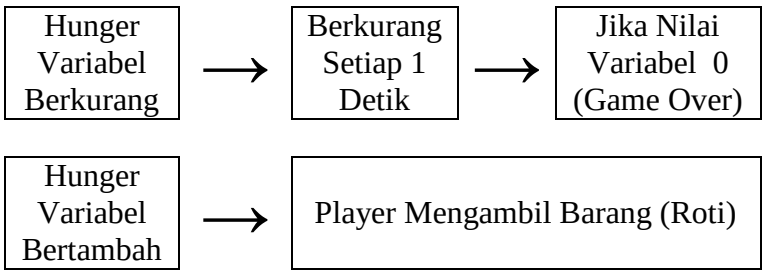
Program ini berjalan pada saat permainan dimulai dan akan terus berjalan. Berikut adalah penjelasan dan model teknis dari program ini :

- Setiap detik nilai lapar *player* akan terus berkurang.
- Misalnya nilai nya 100 akan terus berkurang setiap detiknya sebanyak 1.
- Maka 100 – 1 = 99 , 99 – 1 = 98 dan seterusnya.
- Nilai ini tidak akan minus, kenapa tidak akan minus karena pada saat nilai sama dengan 0 maka akan *game over*.

Bagaimana supaya nilai lapar *player* bertambah, supaya nilai nya bertambah adalah dengan mengambil barang di jalanan (di dalam game barangnya berupa roti). Berikut tahapannya.

- Pertama user harus menggerkan player ke barang terdekat.

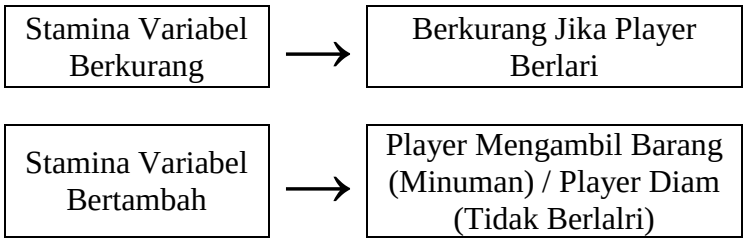
- ❖ Tabaraklah barang tersebut oleh player
- ❖ Maka nilai nya akan bertambah, misal dari 70 menjadi 100.



Gambar IV.7 Model Teknis Pemograman Stamina System

IV.2.2.1.6.2 *Stamina System*

Program ini hampir sama dengan yang *hunger system* bedanya hanya pada barang dan kalau pada program variabel nya berbeda dan pada saat variabel sama dengan 0 tidak akan mati.



Gambar IV.7 Model Teknis Pemograman Stamina System

IV.2.2.1.7 *Inventory System*

Program ini dibuat agar terlihat lebih nyata karena pada saat pendaftaran para mahasiswa biasanya membawa barang, maka dibuatlah sebuah penyimpanan (*inventory*). Di program ini player juga bisa membuang benda yang diambil dengan menekan tombol di keyboard yaitu left ctrl. Berikut adalah penjelasan dan model teknis dari program ini :

- ❖ *User* menggerakan *player* ke barang terdekat dan tabraklah barang tersebut.
- ❖ Maka pada penyimpanan (*inventory*) ada sebuah gambar yang diubah menjadi gambar barang yang di tabrak.

- Untuk mengeluarkan lagi barang tersebut maka *user* harus menekan tombol left ctrl.
- Mengarahkan mouse ke gambar barang yang akan dikeluarkan
- Klik kiri pada gambar tersebut.
- Maka barang akan keluar lagi.

Gambar IV.8 Model Teknis Pemograman Inventory System

IV.2.2.1.8 *Player Objective (Quest)*

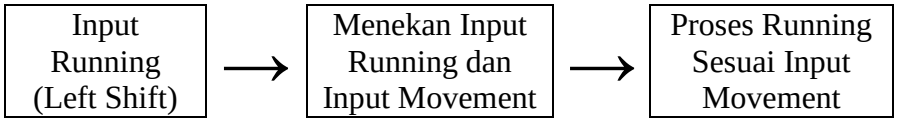
Program ini dibuat untuk navigasi apa yang harus dilakukan pada game ini, misalnya harus ke gedung fti, harus bawa barang terlebih dahulu dan lain lain. Berikut adalah model teknis dari program ini :

Gambar IV.9 Model Teknis Pemograman Player Objective

IV.2.2.1.9 *Player Running*

Program ini dibuat agar player bisa berlari. Program ini menggunakan tombol di keyboard yaitu left shift. Bagaimana tahapannya berikut penjabarannya.

- *User* harus menekan tombol left shift dan *input movement*.
- Maka animasi *player* akan berubah dari animasi berjalan ke animasi berlari.
- *Player* akan berlari sesuai *input movement* (*input movement* sudah dijelaskan pada bagian *movement player*).



Gambar IV.10 Model Teknis Pemograman Runnig System

IV.2.2.1.10 *Main Menu*

Program ini dibuat untuk main menu, disini user bisa memilih beberapa *button*, dan satu *button* ada satu program, berikut adalah *button-button* yang ada di *main manu* dan model teknis dari program ini :

- *Start Game*

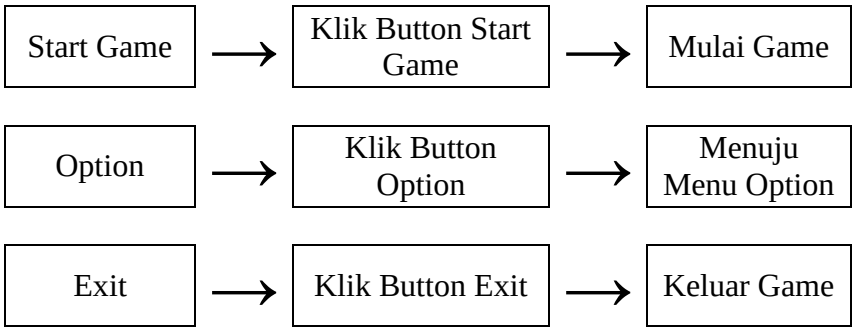
Jiika *user* klik *button* yang ada tulisan "Start Game" nya maka akan memulai permainan.

- *Option*

Jika *user* menekan *button* ini maka akan menyembunyikan *button* Start Option Dan Exit dan akan menampilkan beberapa *button* yaitu *Reset Save*, *Distance –*, *Distance +*, dan *exit*. Dimana *reset save* berperan untuk mereset *player objective,* i untuk mengurangi jarak, *distance+* untuk menambah jarak, dan *exit* untuk kembali ke menu utama.

- *Exit*

Jika *user* menekan *button* ini maka akan keluar dari aplikasi *game* atau kembali ke *desktop*.



Gambar IV.11 Model Teknis Pemograman Main Menu

IV.2.2.2 Log Penyelesaian Program

Nama	Selesai													
	Agustus-September													
	26	27	28	29	30	31	1	2	3	4	5	6	7	8
Movement Player														
Change Camera Position														
Dialogue Player														
Open Door														
Open Level														
Energy System														
Inventory System														
Player Objective														
Player Running														
Main Menu														

Tabel IV.2 Tabel Penyelesaian Tiap Modul Program

IV.3 Pelaporan Hasil Kerja Praktek

Proses pelaporan hasil kerja praktek dilakukan pada tanggal yang diminta oleh perusahaan. Pelaporan hasil kerja praktek ini dilakukan melalui presentasi di hadapan salah satu staff perusahaan. Pelaporan hasil kerja praktek dilakukan pula dengan pembuatan laporan kerja praktek.