BAB 2
LANDASAN TEORI

2.1 Auto Chess

Auto Chess merupakan sebuah permainan bergenre *strategy* dan berjenis *board game* yang dapat dimainkan pada *platform* android. Game ini merupakan costume game dan mode untuk game Dota Auto Chess yang merupakan *mod* pada permainan Dota 2[1]. Dalam permainan ini mengharuskan pemain bertahan selama mungkin dalam permainan hingga semua pemain lawan kehabisan HP(*Hit Point*). Pada permainan ini, pemain diharuskan bertarung dengan pemain lain, dan menghadapi rintangan yang muncul pada *stage* tertentu. Pada setiap stage pemain diharuskan melakukan pembelian bidak karakter mengunakan koin yang diberikan pada setiap *stage*nya, melakukan *upgrade* bidak karakter, melakukan *upgrade* kapasitas maksimal bidak pada arena, menyesuaikan bidak karakter dengan musuh yang dihadapi, hingga menjual bidak yang dimiliki jika diperlukan tambahan koin.

Gambar 2.1 Auto Chess Game play

2.1.1 Game Play

Pada permainan ini, para pemain diharuskan menyusun strategi pada setiap stagenya, dan setiap stage dibagi kedalam tiga fase utama yaitu adalah sebagai berikut.

1. Fase Pembelian Bidak

Fase pembelian bidak merupakan fase pertama dalam permainan, pada fase ini pemain dapat membeli bidak yang muncul pada *pop up* game, dimana karakter bidak yang muncul merupakan karakter bidak yang telah dilakukan pengacakan oleh sistem, sehingga bidak yang muncul selalu berbeda setiap stagenya.

Gambar 2.2 Fase Pembelian

2. Fase Line Up

Fase ini merupakan fase dimana pemain melakukan pengaturan komposisi bidak yang akan dimainkan pada saat *battle.* pada fase ini pemain dapat menganti komposisi bidak yang dimainkan sebelumnya dengan bidak lain yang dimiliki sehingga dapat meningkatkan kesempatan memenangkan permainan.

Gambar 2.3 Fase Line up

3. Fase Upgrade

Fase ini merupakan fase melakukan upgrade bidak yang dapat digunakan dalam arena, proses upgrade kapasitas bidak ini sendiri membutuhkan point yang didapatkan setelah melakukan pertandingan dalam setiap stagenya. Pada dasarnya proses ini dilakukan otomatis oleh sistem game setiap memasuki stage baru, namun pemain dapat melakukan percepatan upgrade kapasitas dengan mengunakan koin dalam pertandingan yang dimiliki.

Gambar 2.4 Fase Upgrade Kapasitas

Berdasarkan pada gambar 2.4 diatas maka contoh perubahan kapasitas pada suatu upgrade kapasitas adalah sebagai berikut.

Gambar 2.5 Contoh Fase pembelian

Setelah tiga fase utama diatas telah dilalui maka masuk kedalam fase utama dalam permainan yaitu fase battle dimana karakter yang telah dipilih player dalam fase line up akan melakukan battle dengan musuh yang telah ditetapkan oleh sistem. Berikut ini adalah contoh fase battle pada sebuah ronde dalam sebuah game.

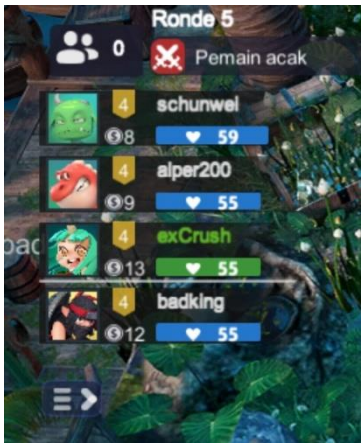
Gambar 2.6 Fase Battle

Fase ini dibagi menjadi dua yaitu fase battle dengan NPC(Non Playable Character) dan juga fase battle dengan pemain lain pada setiap ronde permainan. Fase Battle dengan NPC terjadi pada ronde 1 dan 2 kemudian selanjutnya terjadi setiap 5 ronde selebihnya player akan melakukan battle dengan player lain.

2.1.2 Game Rules

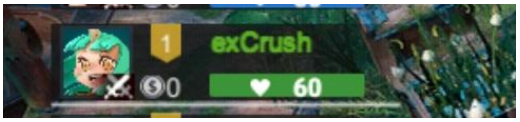
Pada permainan ini terdapat beberapa aturan dalam permainan sebuah pertandingan terdiri aturan aturan dasar yaitu sebagai berikut.

1. Banyaknya pemain pada sebuah pertandingan yaitu sebanyak 4 orang pemain.



Gambar 2.7 Banyak Pemain

2. Kondisi awal pertandingan, setiap pemain akan diberikan hit poin sebanyak 60 HP, 1 koin, dan 1 exp.



Gambar 2.8 Kondisi Awal Pertandingan

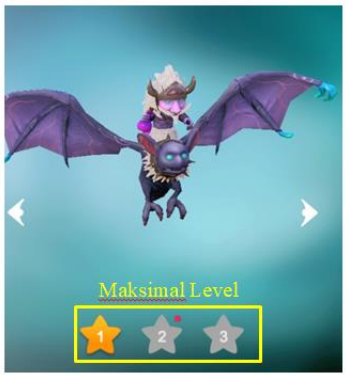
3. Pada fase pembelian, setiap pemain dapat melakukan pembelian karakter maksimal sebanyak 5 karakter dalam satu fase pembelian.

Gambar 2.9 Fase Pembelian

4. Level karakter akan naik jika terdapat 3 karakter dengan level yang sama, proses kenaikan level dilakukan secara otomatis dengan mengabungkan 3 karakter menjadi 1 karakter dengan level yang baru.

Gambar 2.10 Proses Menaikan Level Karakter

5. Maksimal level yang dimiliki oleh setiap karakter yaitu 3 level.



Gambar 2.11 Maksimal Level Karakter

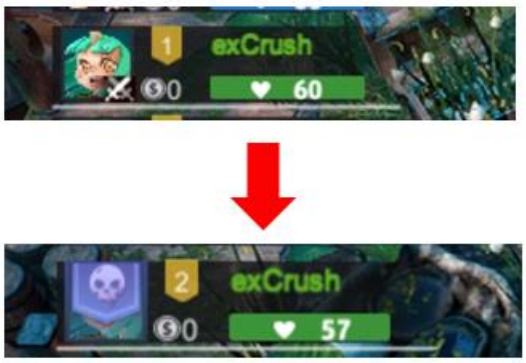
6. Banyak karakter yang dapat digunakan dalam arena adalah sesuai dengan level kapasitas arena yang dimiliki.

Gambar 2.12 Kapasitas Arena

7. Setiap ronde pemain akan mendapatkan Exp kapasitas arena sebesar 1 Exp. Player juga dapat membeli Exp dengan mengunakan koin yang dimiliki dengan harga 4 koin untuk 4 Exp. Jika Exp sudah mencukupi, maka level kapasitas akan naik sebanyak 1 level.

Gambar 2.13 Kapasitas Exp

8. Pemain yang kalah dalam fase battle akan dikenakan pengurangan HP yang dimiliki sebesar (jumlah_karakter_diarena * total_level_karakter_diarena).



Gambar 2.14 Pengurangan HP

9. Pemain dengan HP = 0, maka player tersebut dinyatakan kalah dan akan meninggalkan permainan. Dan pemain yang menang merupakan pemain terakhir yang bertahan dalam pertandingan.

Gambar 2.15 Kondisi Kalah Dalam Pertandingan

2.1.3 Karakter

Pada permainan ini, karakter yang dapat digunakan dalam permainan ini yaitu berjumlah 65 karakter. Setiap karater memiliki sifat, karakteristik, dan cara bertarung yang berbeda dimana Setiap bidak pada permainan ini memiliki tiga atribut utama yaitu sebagai berikut.

1. Atribut karakter

   Atribut karakter terdiri dari 3 atribut yaitu ras dan kelas, kualitas dan harga karakter.

2. Atribut dasar

   Atribut dasar merupakan atribut dasar dari sebuah karakter terdiri dari 6 atribut yaitu *attack, armor, hit point, attack speed, attack area*, dan *magic resistant*.

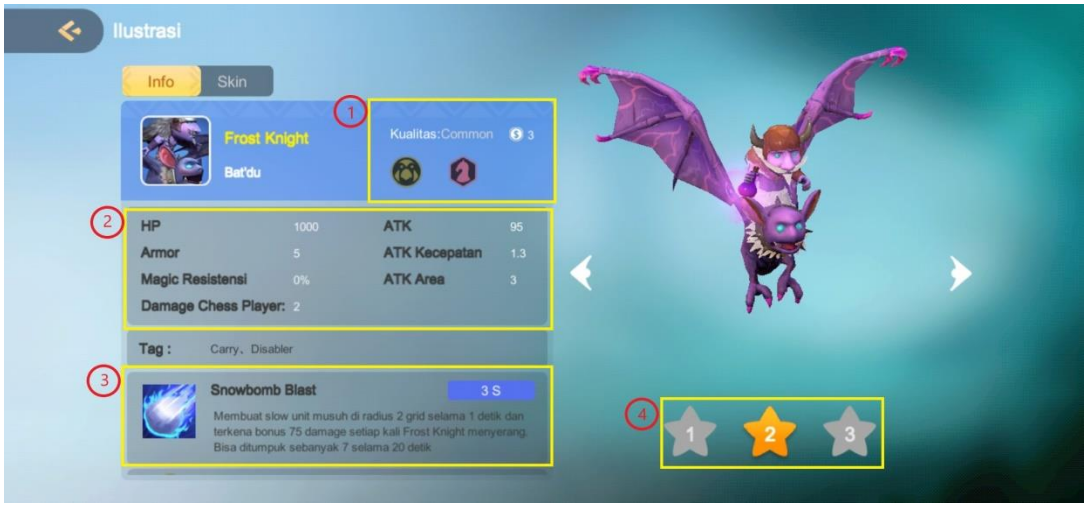
3. Atribut skill

   Merupakan atribut unit untuk setiap karakter dengan efek, *cold down* dan area yang berbeda beda.

4. Atribut level

   Merupakan atribut level yang dimiliki oleh setiap karakter pada permainan Auto Chess.

Berikut ini adalah contoh penerapan empat atribut utama dalam sebuah karate bidak dalam game Auto Chess.



Gambar 2.16 Atribut Karakter

Berdasarkan gambar 2.16 diatas maka atribut setiap karakter dapat dijelaskan secara detail pada tabel dibawah ini.

Tabel 2. 1 Detail Atribut Setiap Karakter

Atribut	Nama	Keterangan
Atribut Level	Level	Level karakter
Atribut Dasar	Attack	Besarnya kerusakan serangan
	Armor	Besarnya ketahanan terhadap serangan bertipe fisik
	Hit Point (HP)	Besarnya hit point yang dimiliki
	Attack Speed	Kecepatan Serangan per detik
	Magic Resistance	Besarnya ketahanan terhadap serangan bertipe magic
	Attack Range	Jarak serangan yang dimiliki
Atribut Skill	Skill Type	Tipe Skill yang dimiliki
	Skill Efect	Tipe kerusakan dari skill
	Skill Area	Rentang area skill
	Skill Duration	Durasi dari skill
	Skill Damage	Besarnya kerusakan dari skill
	Skill Cooldown	Lamanya waktu aktif skill untuk skill berikutnya
	Skill Chance	Kesempatan aktifnya skill
Atribut karakter	Quality	Tipe kualitas yang dimiliki
	Class	Tiper kelas yang dimiliki
	Race	Tiper ras yang dimiliki

Setiap karakter dalam game memiliki atribut karakter berbeda, atribut ini terdiri dari kualitas, ras dan kelas. Kualitas merupakan sebuah atribut yang mengelompokan karakter berdasarkan kekuatan dan tingkatan karakter, Semakin tinggi kualitas yang dimiliki semakin rendah pula kesempatan karakter tersebut muncul pada fase pembelian. Karakter dalam game ini dibagi menjadi 5 jenis berdasarkan kualitas yaitu.

1. Legendaris
2. Mistis
3. Langka
4. Tidak Umum
5 Umum

Ras merupakan sebuah atribut yang mengelompokan karakter berdasarkan jenis karakter. Dalam permainan ini, karakter yang tersedia terbagi mernjadi 12 ras yaitu sebagai berikut.

Tabel 2. 2 Daftar Ras

No	Nama Ras	Simbol
1	Iblis	
2	Dewa	
3	Dragon	
4	Gletser	
5	Egeris	
6	Bersayap	
7	Goblin	
8	Roh	
9	Laut	
10	Kurcaci	
11	Mahluk Buas	

No	Nama Ras	Simbol
12	Manusia	
13	Suku Manusia Gua	
14	Kira	

Kelas merupakan sebuah atribut yang mengelompokan karakter berdasarkan jenis, cara dan efek bertarung. Dalam permainan ini, karakter yang tersedia terbagi mernjadi 12 Kelas yaitu sebagai berikut.

Tabel 2. 3 Daftar Kelas

No	Nama Kelas	Simbol
1	Tukang Sihir	
2	Petarung	
3	Dukun	
4	Pendeta	
5	Cenayang	
6	Pemburu	
7	Meka	
8	Balian	
9	Ksatria	

No	Nama Kelas	Simbol
10	Syaman	
11	Pembunuh Bayaran	
12	Penyihir	

Setiap Ras dan Kelas memiliki efek bonus yang disebut sinergi. Setiap efek sinergi hanya akan aktif ji =-[ka memenuhi persyaratan jumlah sinergi untuk setiap kelas dan ras memiliki sarat serta efek yang berbeda beda setiap karakter. Berikut ini merupakan gambaran efek sinergi dalam sebuah permainan.

Gambar 2. 17 Sinergi Ras dan Kelas

Berdasarkan daftar kualitas, ras pada tabel 2.1 dan juga daftar kelas 2.1 maka daftar karakter yang dapat digunakan dalam permainan ini adalah sebagai berikut.

Tabel 2. 4 Daftar Karakter

No	Nama	Kualitas	Ras	Kelas
1	Shadow Devil	Langka	Iblis	Cenayang
2	Lord of Sand	Langka	Hewan buas	Pembunuh bayaran
3	Thunder spirit	Langka	Roh	Dukun

No	Nama	Kualitas	Ras	Kelas
4	Fortune teller	Langka	Gletser	Pendeta
5	Argali knight	Langka	Manusia	kesatria
6	Flaming wizard	Langka	Manusia	Dukun
7	Poisonous worm	Langka	Hewan buas	Cenayang
8	Fallen witcher	langka	Iblis	Penyihir
9	Warpwood sage	Langka	bersayap	Balian
10	Wind ranger	Langka	Bersayap	Pemburu
11	Shadowcrawler	Langka	Bersayap	Pembunuh bayaran
12	Warewolf	Langka	Manusia	Petarung
13	Grand herald	Tidak Umum	Dewa	Tukang sihir
14	Evil knight	Langka	Egersis	Kesatria
15	Venom	Langka	Naga	Pembunuh bayaran
16	Dwarf sniper	Langka	Kurcaci	Pemburu
17	Phantom queen	Tidak Umum	Iblis	Pembunuh bayaran
18	Water spirit	Tidak Umum	Roh	Pembunuh bayaran
19	Ripper	Tidak Umum	Goblin	Mekka
20	Shining archer	Tidak Umum	Bersayap	Pemburu
21	Frost knight	Umum	Gletser	Kesatria
22	Egersis ranger	Umum	Egersis	Pemburu
23	Stone spirit	Umum	Roh	Petarung
24	Heaven bomber	Umum	Goblin	Mekka
25	Soul breaker	Umum	Goblin	Pembunuh bayaran
26	God of war	Umum	Dewa	Petarung
27	Winter chiropteran	Umum	Naga	Dukun
28	Lightblade knight	Tidak umum	Bersayap	Kesatria
29	Wisper seer	Tidak umum	Bersayap	Balian
30	Shining dragon	Tidak umum	Naga	Dukun
31	Desperate doctor	Tidak umum	Gletser	Cenayang
32	Abyssal guard	Tidak umum	Laut	Petarung
33	Abyssal crawler	Tidak umum	Laut	Pembunuh bayaran

No	Nama	Kualitas	Ras	Kelas
34	Strange egg	Legendaris	Bersayap	-
35	Helicopter	Legendaris	Kurcaci	Mekka
36	Devastator	Legendaris	Goblin	Mekka
37	Dark spirit	Legendaris	Roh	Cenayang
38	Tortola elder	Mistis	Manusia	Dukun
39	Soul reaper	Mistis	egersis	cenayang
40	Doom arbiter	Mistis	Iblis	Petarung
41	Grimtouch	Mistis	Iblis	Tukang sihir
42	Rouge guard	Legendaris	Iiblis	Petarung
43	Unicorn	Umum	Hewan buas	Balian
44	Ogre mage	Umum	Kira	Dukun
45	Sky breaker	Umum	Goblin	Mekka
46	Tsunami stalker	Legendaris	Laut	Pemburu
47	Bliht sorcerer	Legendaris	Egersis	Dukun
48	God of thunder	Legendaris	Dewa	Dukun
49	Egersis prophet	Legendaris	Egersis	Cenayang
50	Redaxe chief	Umum	Suku gua	Petarung
51	Tusk champion	Umum	Hewan buas	Petarung
52	Taboo witcher	Umum	Bersayap	Penyihir
53	Difector	Umum	Gletser	Syaman
54	Umbra	Tidak umum	Naga	Pemburu
55	The source	Tidak umm	Manusia	Dukun
56	Skull hunter	Tidak umum	Suku gua	Pemburu
57	Swordman	Tidak umum	Suku gua	Petarung
58	Hell knight	Tidak umum	Iblis	Kesatria
59	Shadow devil	Langka	Iblis	Cenayang
60	Berserker	Mistis	Gletser	Petarung
61	Shining assassin	Mistis	Bersayap	Pembunuh bayaran
62	Razorclaw	Mistis	Hewan buas	Balian
63	Dragon knight	Mistis	Naga	Keastria

No	Nama	Kualitas	Ras	Kelas
64	Siren	Mistis	Laut	Pemburu
65	Strom shaman	Mistis	Suku gua	Syaman

2.2 *Reinforcement Learning* (RL)

Reinforcement Learning(RL) atau pembelajaran penguatan merupakan jenis ketiga dalam *machine learning* setelah metode *supervised* dan *unsupervised*. RL merupakan sebuah metode *pelatihan* yang menggunakan konsep *"what to do – how to map situation to action – so as to maximaze a numrical reward signal"*[4], yang bisa diartikan apa yang harus dilakukan, bagaimana cara memetakan situasi untuk melakukannya, sehingga mendapatkan hadiah (*reward*) semaksimal mungkin. RL juga sering disebut sebagai konsep *pelatihan* "*reward or punishment".* Secara umum proses dalam RL melibatkan tiga model dasar yaitu:

a) Agent

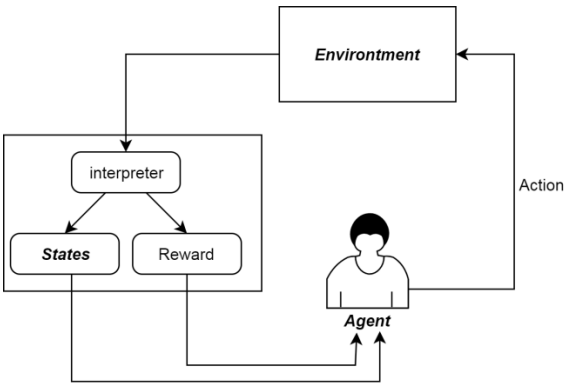
Agent adalah sebuah object atau individu yang akan melakukan aksi pada sebuah *environment. Action* pada agen sendiri didasari pada apa yang bisa ia lakukan pada sebuah *environment* seperti berjalan, menunduk, melompat, mengambil, dan berbelok.

b) Environment

Environment atau lingkungan merupakan sebuah ruang lingkup pada suatu kasus yang akan diselesaikan *agent*, yang berisikan berbagai permasalahan atau rintangan yang akan dihadapi *agent* [10].

c) States

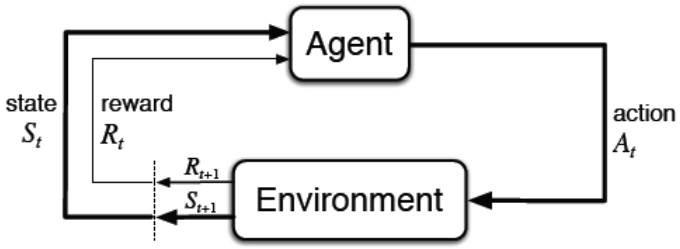
States merupakan kondisi kondisi yang dihadapi *agent*, kondisi kondisi inilah yang kemudian menjadi dasar pembelajaran dan keputusan pada *agent*. Hubungan antara ketiga model dasar di atas disebut *Markov Decision Proccess* (MDP)[4]. secara umum dapat digambarkan pada seperti di bawah ini.



Gambar 2. 18 *Markov Decision Proccess* (1)

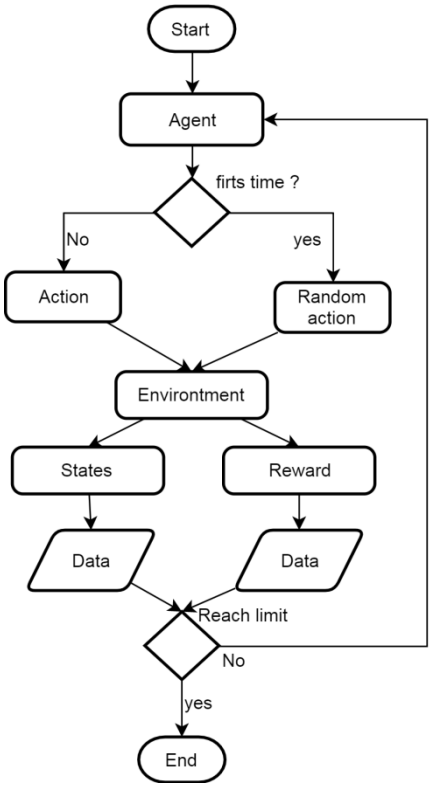
2.2.1. *Markov Decision Processes* (MDP)

Markov Decision Proccess (MDP) merupakan sebuah model formulasi matematika klasik dalam pembuatan dan pengambilan keputusan berurutan. dimana *action* atau tindakan tidak hanya mempengaruhi *reward* yang diterima tetapi juga *states* atau kondisi yang akan dihadapi berikutnya, sehinga berimbas kepada *reward* yang akan diterima dimasa yang akan datang[10]. Model MDP secara detail dapat digambarkan pada gambar di bawah ini.



Gambar 2. 19 Markov Decision Proccess (2)

Berdasarkan keterhubungan tiga model dasar yang telah dijelaskan pada gambar 2.19 maka secara umum proses yang terjadi dapat digambarkan seperti flowchart dibawah ini.



Gambar 2. 20 MDP Flowchart

Pada gambar 16, *agen* dan *environment* selalu melakukan interaksi pada setiap urutan waktu diskrit $t = 0,1,3,4,5,. . . .$ [2] di setiap waktu t. Pada setiap langkah waktu t, *agen* menerima beberapa representasi atau gambaran kondisi *environment* $S_t \in S$ dan menjadikan kondisi *environment* tersebut sebagai dasar pengambilan sebuah *action* $A_t \in A(s)$.[3] . Pada langkah berikutnya, sebagai bagian konsekuensi agen akan menerima sebuah *reward* $\mathbf{R}_{t+1}\ \mathbf{R} \in \mathbf{R}$, dan memberitahu agent bahwa ia berada pada *state* baru S_{t+1}.[4] . Proses tersebut akan membentuk sebuah urutan atau lintasan sebagai berikut [14]:

$$S_0, A_0, R_1, S_1, A_1, R_2, S_2, A_2, R_2, \ldots . \quad (2.1)$$

Dalam MDP sendiri terdiri dari dua jenis yag pertama yaitu *Finite Markov Decision Proccess* (*finite* MDP), dan *Infinite Markov Decision Proccess* (*infinite* MDP) berikut merupakan penjelasan singkat dua jenis MDP tersebut[14]:

1. *Infinite Markov Decision Proccess* (*infinite* MDP)

Infinit MDP merupakan sebuah MDP dimana kondisi *set of state* , *action*, dan *reward* (***S, A,*** dan ***R***) tidak dibatasi, artinya (***S, A,*** dan ***R***) dapat berkembang sesuai

dengan kondisi yang ada, biasanya jenis ini digunakan pada sebuah sistem dengan kondisi *environment* yang berubah ubah.

2. *Finite Markov Decision Proccess* (*finite* MDP),

Finite MDP merupakan sebuah MDP dimana kondisi *set of state , action,* dan *reward* (***S, A,*** dan ***R***) dibatasi, artinya (***S, A,*** dan ***R***) bernilai tetap atau didefinisikan terlebih dahulu.

Metode pengambilan keputusan atau *action* untuk setiap *state* dalam RL sendiri bisa dilakukan dengan berbagai cara salah satunya yaitu *ε-greedy*. Pengambilan keputusan sebuah *action* dalam RL terbagi menjadi dua yaitu:

a) Exploration

Exproration merupakan proses pengambilan keputusan untuk setiap *action* yang didapatkan secara acak(*random*). Dimana keputusan ini akan diambil berdasarkan probabilitas $\boldsymbol{\varepsilon}$.

b) Exploitation

Exploitation merupakan proses perngambilan keputusan untuk setiap *action* yang didapatkan dari hasil seleksi *value function* terbesar. Dimana keputusan ini akan diambil berdasarkan probabilitas (1 - $\boldsymbol{\varepsilon}$**)**. Perhitungan *epsilon geedy strategy* ini dapat dihitung menggunakan rumus sebagai berikut.

$$\boldsymbol{E = r - e} \tag{2.1}$$

Dimana :

r = nilai *random* (0 < r < 1)

e = nilai epsilone (0 < e < 1)

E = explorasi

2.2.2. *Dynamic Programing* (DP)

Dynamic programing (DP) merupakan merupakan sebuah metode utilitas terbatas dalam RL. DP sendiri cocok digunakan pada RL dengan asumsi lingkungan atau *Environtment* yang jelas ataupun sempurna, sehingga tidak perlu menggunakan rumus perhitungan matematika dalam proses analisa *states* yang terjadi. *Dynamic programing* (DP) sendiri mengacu kepada kompulan algoritma yang dapat digunakan untuk menciptakan *policy* (kebijakan) yang optimal

berdasarkan model lingkungan atau *state* sehingga *action* yang dihasilkanpun optimal. *Policy* sendiri secara umum dapat digambarkan dengan rumus sebagai berikut.

$$\boldsymbol{\pi : A\ x\ S\ \rightarrow\ [0,1]} \tag{2.2}$$

$$\boldsymbol{\pi(a,s) = \mathbf{Pr}(a_t = a \mid s_t = s)} \tag{2.3}$$

Dimana : s = *states*

a = *action*

r = *reward*

Pr = *probability*

Gagasan utama DP dalam RL sendiri adalah untuk memanfaatkan *function value* atau fungsi nilai untuk mengatur dan menyusun pencarian kebijakan yang baik sehingga keputusan yang diambil merupakan keputusan yang optimal. Dimana *function value* (v_* atau q_*) didapatkan dengan mengunakan rumus *Bellman Opimality Equation* sebagai berikut[14] .

$$\boldsymbol{v_*(s)} \quad = \frac{\boldsymbol{MAX}}{\boldsymbol{a}}\ \boldsymbol{E\ [R_{t+1} + \gamma v_*(S_{t+1}) \mid S_t = s, A_t = a]} \tag{2.4}$$

$$= \frac{\boldsymbol{MAX}}{\boldsymbol{a}}\ \boldsymbol{\sum_{s',r} p(s', r \mid s, a)[r + \gamma v_*(s')\ ]} \tag{2.5}$$

atau

$$\boldsymbol{q_*(s,a)} \quad = \boldsymbol{E\ [R_{t+1} + \gamma\ max_{a'} q_*\ (S_{t+1}, a') \mid S_t = s, A_t = a\ ]} \tag{2.6}$$

$$= \boldsymbol{\sum_{s',r} p(s', r \mid s, a)[r + \gamma q_*(s', a')\ ]} \tag{2.7}$$

Dimana : s = *states*

a = *action*

r = *reward*

p = *probability*

1.2.3 QLearning (QL)

QLearning (QL) merupakan algoritma kembangan dari algoritma *Temporal Defence Learning* (TD). QL dalam MDP berfungsi menemukan kebijakan yang optimal untuk memaksimalkan *reward* yang diperoleh secara

berturut turut dengan melakukan update terhadap *state-value function*[4]. Secara umum *state-value function* dapat diperoleh dengan rumus berikut.

$$\boldsymbol{q_\pi}(\boldsymbol{s},\boldsymbol{a}) = \boldsymbol{E_\pi}[\boldsymbol{G_t} \mid \boldsymbol{S_t} = \boldsymbol{s}, \boldsymbol{A_t} = \boldsymbol{a}] \quad (2.8)$$

Dimana :

E_π = *Expected policy*

G_t = *Ecpected return*

S_t = *State* pada time ke-t

A_t = *Action* pata time ke-t

a = *Action*(line up)

s = *State*

Berdasarkan rumus di atas expected return (G_t) diperoleh dengan melakukan kalkulasi reward pada state dan action yang sama. Proses perolehan nilai ini dapat ditumuskan dengan rumus sebagai berikut.

$$\boldsymbol{G_t} = \boldsymbol{R_{t+1}} + \boldsymbol{R_{t+2}} + \boldsymbol{R_{t+3}} + \cdots + \boldsymbol{R_T} \quad (2.9)$$

Dimana :

G_t = *Expected Return*

R = *Reward*

t = *Time (step)*

Setelah *state-value function* diketahui maka kemudian dilakukan perhitungan performasi agen pada setiap *state* yang sama. Perhitungan performasi ini dihitung dengan rumus optimasi Bellman yaitu sebagai berikut.

$$\boldsymbol{q_*}(\boldsymbol{s},\boldsymbol{a}) = \boldsymbol{E}[\boldsymbol{R_{t+1}} + \boldsymbol{\gamma}\,\mathbf{max}_{\boldsymbol{a'}}\,\boldsymbol{q_*}(\boldsymbol{s'},\boldsymbol{a'})] \quad (2.10)$$

Dimana :

$q_*(s,a)$ = *Optimal value function*

E = *Ecpected*

R_{t+1} = *Reward* yang diterima pada time berikutnya

$\max_{a'} q_*(s',a')$ = *Maximal reward* yang diterima

γ = *Hyperparameter* (gamma)

QL mengidentifikasi *policy* yang ada pada MDP, dan memberikan keputusan yang optimal dengan waktu ekplorasi yang tidak terbatas. QL secara umum dapat digambarkan sebagai rumus berikut .

$$\boldsymbol{q^{new}(s,a) = (1-\alpha) - q(s,a) + \alpha\left(R_{t+1} + \gamma \max_{a'} q_*(s',a')\right)}$$

$$\boldsymbol{q^{new}(s,a) = (1-\alpha) - q(s,a) - \alpha(q_*(s,a))} \quad (2.11)$$

Dimana :

α = *Learning rate*

$q(s,a)$ = nilai Q-value lama

$q_*(s,a)$ = Nilai Optimasi Belman

2.3 *Unified Modeling Language* (UML)

Unified Modeling Language (UML) adalah bahasa pemodelan untuk menspesifikasikan, memvisualisasikan, membangun dan mendokumentasikan artifak-artifak dari sistem. UML digunakan dalam pengembangan sistem perangkat lunak yang menggunakan pendekatan berorientasi objek [11]. Adapun yang menjadi pegangan dalam pembuatan UML ini berdasarkan Ebook oleh Rules Miles.

2.4 Use Case Scenario

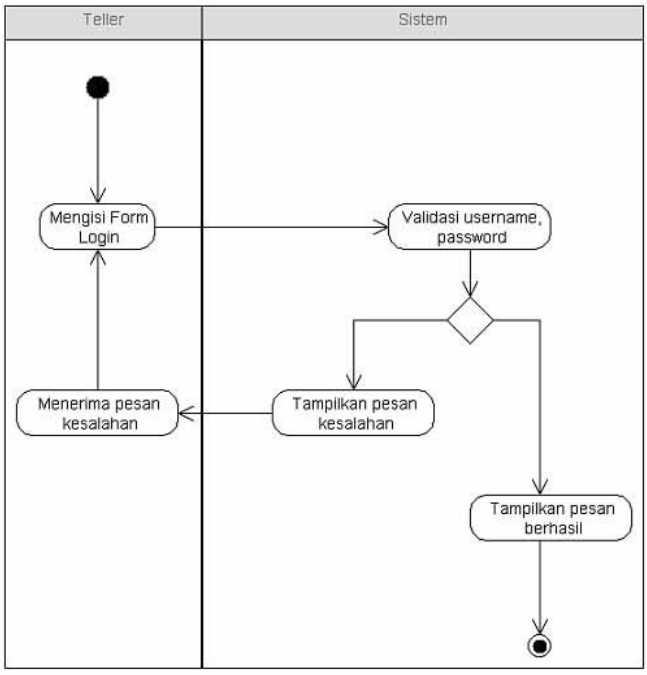
Use-case scenario dijelaskan secara tekstual dalam beberapa format tergantung kebutuhannya, yaitu singkat (brief), informal (casual), atau lengkap (fully dressed), yang dapat dijelaskan dalam satu atau dua kolom. Pada penelitian ini, digunakan format sebagai berikut.

1. Nama Use Case (Use case name)
2. Persyaratan Terkait (Related Requirements), kondisi spesifik yang harus terpenuhi sebelum use-case dieksekusi oleh aktor.
3. Tujuan (Goal in Context), penggunaan use case dan menjelaskan mengapa use case ini menjadi penting digunakan.
4. Kondisi Awal (Preconditions), kondisi awal sebelum use case dieksekusi.
5. Kondisi Akhir Berhasil (Successful End Condition), kondisi ketika use case berhasil dieksekusi.
6. Kondisi Akhir Gagal (Failed End Condition), kondisi ketika use case gagal dieksekusi.
7. Aktor Utama (Primary Actors), Aktor utama yang menggunakan use case.

8. Aktor Sekunder (Secondary Actors), Aktor sekunder/lainnya yang menggunakan use case.
9. Pemicu (Trigger), pemicu oleh aktor sehingga use case dieksekusi.
10. Alur Utama (Main Flow), penjelasan terkait alur utama dari use case apabila berjalan secara normal.
11. Ekstensi (Ekstensions), yaitu jalur alternatif dari interaksi yang terjadi antara aktor dan sistem yang mencakup percabangan (pilihan) maupun skenario yang gagal sehingga tujuan aktor tidak terpenuhi.

2.5 Activity Diagram

Activity diagram adalah diagram flowchart yang diperluas yang menunjukkan aliran kendali satu aktivitas ke aktivitas lain berdasarkan use case diagram. Berikut adalah contoh dari activity diagram pada gambar dibawah ini.



Gambar 2. 21 Contoh Activity Diagram

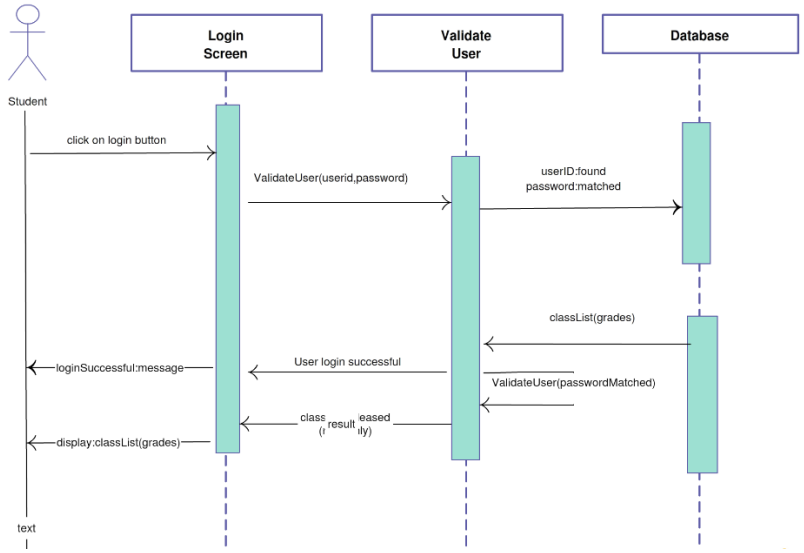
2.6 Class Diagram

Class diagram merupakan inti dari setiap sistem berorientasi objek, oleh karena itu kaitannya sangat erat dengan use-case diagram. Class diagram bertujuan untuk menjelaskan berbagai jenis objek dan hubungannya dengan kelas lainnya

yang dimiliki oleh sistem. Hal ini lah yang menjadikan class diagram memiliki dua macam informasi, yaitu informasi tentang kondisi awal objek dan bagaimana berperilaku dalam lingkungannya.

2.7 Sequence Diagram

Squence diagram atau dikenal juga sebagai diagram interaksi bertujuan untuk memodelkan interaksi secara runtime antara bagian – bagian tertentu pada sistem yang membentuk alur perpindahan sebuah objek. Diagram ini akan meyampaikan urutan dari setiap interaksi pada suatu proses atau bagian – bagian sistem[12]. Dengan menampilkan apa yang menjadi pemicu dari sebuah proses dan menunjukkan informasi lain berupa peristiwa dalam suatu interaksi.Berikut adalah contoh dari sequence diagram pada gambar dibawah ini.



Gambar 2. 22 Contoh Sequence Diagram