

RANCANG BANGUN GAME SERVER ONLINE PADA PERMAINAN STRATEGY KERAJAAN MAJAPAHIT

SKRIPSI

Diajukan untuk memenuhi persyaratan memperoleh gelar Sarjana Komputer



Disusun Oleh :

ADITYO NUGROHO

NIM. 0710683036

KEMENTERIAN PENDIDIKAN DAN KEBUDAYAAN

UNIVERSITAS BRAWIJAYA

PROGRAM TEKNOLOGI INFORMASI DAN ILMU KOMPUTER

PROGRAM STUDI INFORMATIKA/ILMU KOMPUTER

MALANG

2014

LEMBAR PERSETUJUAN

**RANCANG BANGUN GAME SERVER ONLINE PADA
PERMAINAN STRATEGY KERAJAAN MAJAPAHIT**

SKRIPSI

Diajukan untuk memenuhi persyaratan memperoleh gelar Sarjana Komputer



Disusun Oleh :

ADITYO NUGROHO

NIM. 0710683036

Telah diperiksa dan disetujui oleh:

Dosen Pembimbing I

Dosen Pembimbing II

Eriq M. Adams J., S.T., M.Kom

NIP. 198504102012 1 2 1001

Wibisono Sukmo Wardhono, S.T., M.T

NIK. 820404 06 1 1 0091



LEMBAR PENGESAHAN

RANCANG BANGUN GAME SERVER ONLINE PADA
PERMAINAN STRATEGY KERAJAAN MAJAPAHIT

Disusun Oleh :

ADITYO NUGROHO

NIM. 0710683036

Skripsi ini telah diuji dan dinyatakan lulus pada tanggal 10 Juli 2014

Penguji I

Denny Sagita R., S.Kom., M.Kom
NIK. 851124 06 1 1 0250

Penguji II

Issa Arwani, S.Kom., M.SC.
NIK. 198309222012121003

Penguji III

Sabriansyah Rizqika A., ST., M.Eng.
NIK. 820809 06 1 1 0084

Mengetahui

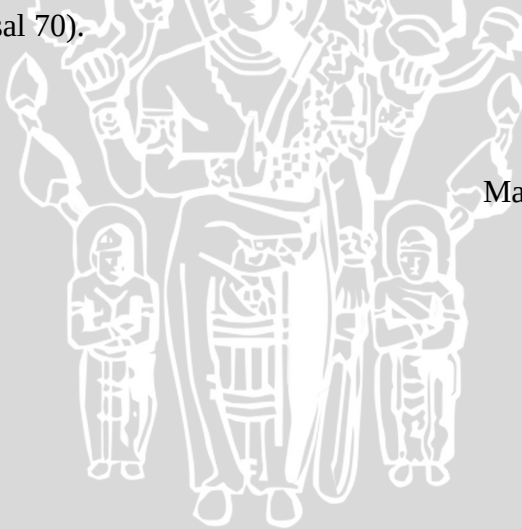
Ketua Program Studi Informatika/Ilmu Komputer

Drs. Mardji, MT.
NIP. 19670801 199203 1 001

**PERNYATAAN
ORISINALITAS SKRIPSI**

Saya menyatakan dengan sebenar-benarnya bahwa sepanjang pengetahuan saya, di dalam naskah SKRIPSI ini tidak terdapat karya ilmiah yang pernah diajukan oleh orang lain untuk memperoleh gelar akademik di suatu perguruan tinggi dan tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis dikutip dalam naskah ini dan disebutkan dalam sumber kutipan dan daftar pustaka.

Apabila ternyata di dalam naskah SKRIPSI ini dapat dibuktikan terdapat unsur-unsur PLAGIASI, saya bersedia SKRIPSI ini digugurkan dan gelar akademik yang telah saya peroleh (SARJANA) dibatalkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku (UU No. 20 Tahun 2003, Pasal 25 ayat 2 dan Pasal 70).



Malang, 21 Juni 2014

Mahasiswa,

Adityo Nugroho

0710683036

KATA PENGANTAR

Dengan menyebut nama Allah SWT yang Maha Pengasih dan Maha Penyayang. Segala puja dan puji syukur bagi Allah SWT karena berkat limpahan rahmat dan hidayah-Nya sehingga penulis dapat menyelesaikan dengan baik Laporan Skripsi yang berjudul, **“RANCANG BANGUN GAME SERVER ONLINE PADA PERMAINAN STRATEGY KERAJAAN MAJAPAHIT”**. Shalawat dan salam semoga selalu tercurah atas junjungan kita Nabi besar Muhammad SAW beserta keluarga dan para sahabat.

Tugas Akhir ini disusun untuk memenuhi sebagian persyaratan memperoleh gelar Sarjana Komputer di Program Studi Teknik Informatika Program Teknologi Informasi dan Ilmu Komputer Universitas Brawijaya Malang. Pada penyelesaian tugas akhir ini, penulis telah mendapat begitu banyak bantuan baik moral maupun materiil dari berbagai pihak. Melalui kesempatan ini, atas bantuan yang telah diberikan, penulis ingin menyampaikan penghargaan sebesar-besarnya serta ucapan terima kasih yang sedalam-dalamnya kepada:

1. Ibunda Herwahjuni, Almarhum Ayahanda Sugondo atas segala nasehat, perhatian, kasih sayang dan kesabarannya dalam membesarkan dan mendidik penulis serta senantiasa berdoa dan memberi dukungan dan semangat.
2. Bapak Eriq M. Adams J. S.T., M.Kom dan Bapak Wibisono Sukmo W, S.T., M.T. selaku dosen pembimbing yang memberikan ilmu dan saran dalam penyusunan laporan tugas akhir ini.
3. Bapak Ir. Sutrisno, M.T selaku dosen pembimbing akademik yang selalu memberikan nasehat pada penulis saat menempuh masa studi.
4. Teman – teman asisten Laboratorium Dasar Pemrograman Komputer Teknik Elektro Universitas Brawijaya.
5. Teman – teman asisten Laboratorium Game Teknik Informatika Universitas Brawijaya.
6. Seluruh pihak yang terlibat baik secara langsung maupun tidak langsung, yang tidak bisa disebutkan satu persatu.

Hanya doa yang bisa penulis berikan, semoga Allah SWT memberikan pahala dan balasan yang berlipat. Penulis menyadari bahwa skripsi ini masih jauh dari sempurna. Oleh karenanya, penulis mengharapkan masukan berupa kritik dan saran yang membangun. Semoga skripsi ini dapat memberikan manfaat bagi penyusun dan pihak lain yang ingin menggunakan.



Malang, 21 Juni 2014

Penulis

ABSTRAK

Adityo Nugroho. 2014. : RANCANG BANGUN GAME SERVER ONLINE PADA PERMAINAN STRATEGY KERAJAAN MAJAPAHIT. Skripsi Program Studi Teknik Informatika, Program Teknologi Informasi dan Ilmu Komputer, Universitas Brawijaya. Dosen Pembimbing : Eriq M. Adams J., S.T., M.Kom dan Wibisono Sukmo Wardhono, ST, MT

Permainan strategy Kerajaan Majapahit ialah sebuah sarana hiburan kompetitif *multiplayer* serta *multi platform* yang sekaligus berfungsi sebagai sarana penarik minat generasi penerus bangsa agar lebih peduli terhadap sejarah warisan leluhur bangsa, khususnya sejarah Kerajaan Majapahit. Dalam permainan, pemain akan berperan sebagai kepala desa suatu wilayah yang berada di bawah sistem pemerintahan Kerajaan Majapahit. Sebagai kepala desa, pemain dituntut untuk berperan aktif dalam mengembangkan, memakmurkan serta menjaga keamanan wilayah yang dipimpinnya.

Perancangan *game server online* pada permainan ini melalui dua tahapan yakni perancangan *game concept design* dan *technical design*. Implementasi algoritma *gameplay* dilakukan sepenuhnya dengan bahasa *query* pada MySQL, sedangkan dalam implementasi *web* akan dikembangkan *PHP based website* yang berfungsi sebagai *interface* atau antarmuka antara *game server online* dengan *compatible game client*. Pengujian perangkat lunak aplikasi memakai metode *white-box testing* dengan pemakaian teknik *basis path testing* pada strategi pengujian unit dan pengujian integrasi serta metode *black-box testing* dengan strategi pengujian validasi. Dari hasil pengujian bisa ditarik kesimpulan bahwa *game server online* telah mampu beroperasi sebagai mana mestinya dalam memenuhi kebutuhan fungsional maupun non fungsional.

Kata Kunci : Permainan *Strategy*, Kerajaan Majapahit, *Game Server Online*

ABSTRACT

Adityo Nugroho. 2014. : *Development Of Online Game Server for Strategic Game of Majapahit Empire. Undergraduate Thesis of Informatic Study Program, Information Technology and Computer Science Program, Brawijaya University, Malang. Advisor : Eriq M. Adams J., S.T., M.Kom and Wibisono Sukmo Wardhono, S.T., M.T.*

Majapahit Empire strategy game is a multiplayer competitive entertainment facilities as well as multi-platform that also functions as a means of towing the interest of future generations to be more concerned about the historical heritage of the nation, especially the history of the kingdom of Majapahit. In the game, players will assume the role as head of the village of an area that is under the governance system of the Majapahit Kingdom. As the head of the village, players are required to play an active role in developing, maintaining prosperity and security of the region lead.

The design of online game servers through two stages of the game design concept and technical design. Implementation of the gameplay algorithm is performed entirely with MySQL query language, while the web implementation will be implemented PHP web-based website that serves as an interface between the online game servers with compatible game clients.

Testing applications using white-box testing methods with the use of the base path technique for unit testing and integration testing and black-box testing methods with validation testing strategies. From the test results can be deduced that the online game server has been able to operate as it should to meet the functional and non-functional requirements.

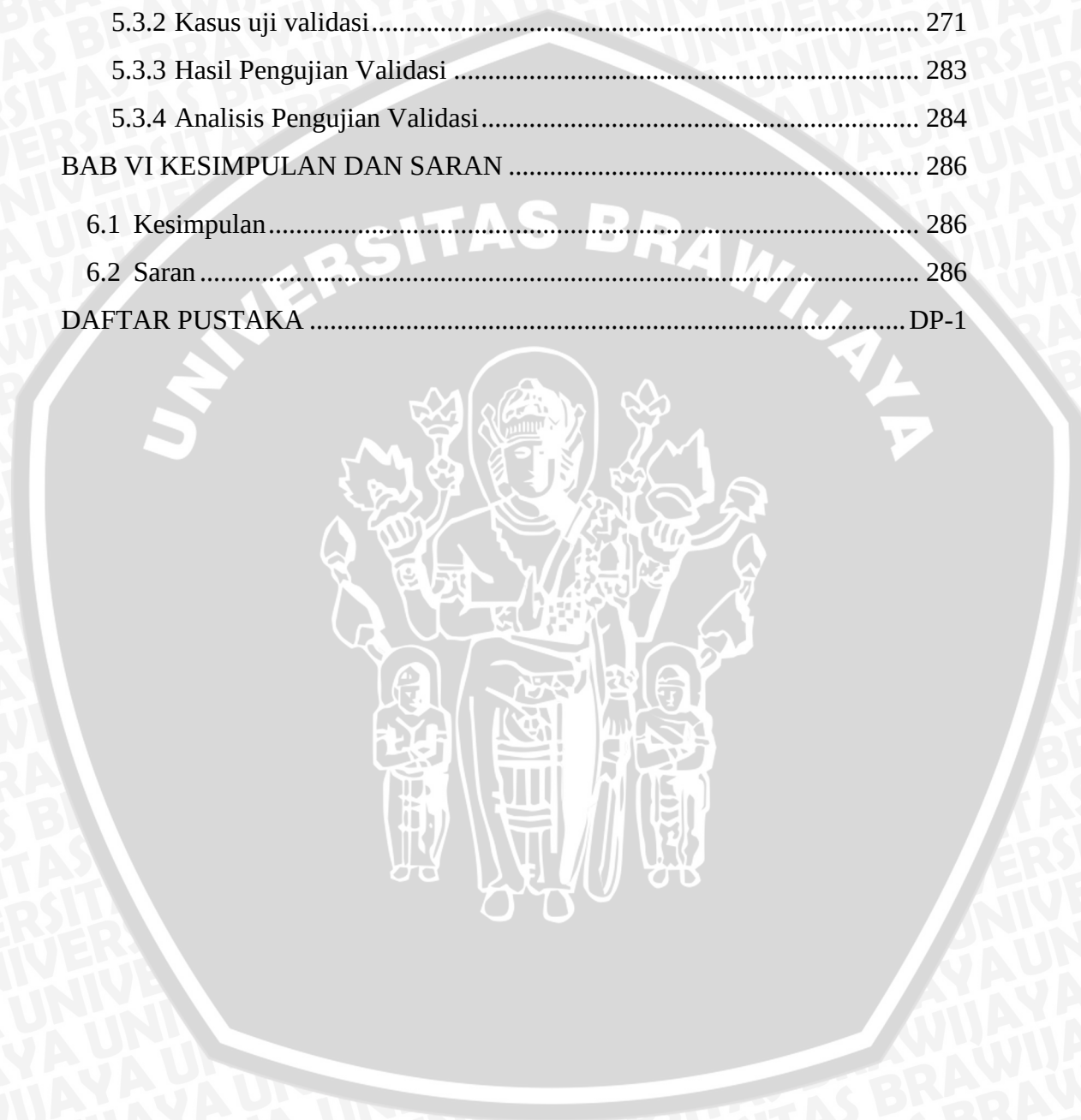
Keywords: Strategy Games, Majapahit Empirer, Online Game Server

DAFTAR ISI

KATA PENGANTAR	i
ABSTRAK	iii
ABSTRACT	iv
DAFTAR ISI	v
DAFTAR GAMBAR	viii
DAFTAR TABEL	x
BAB I PENDAHULUAN	1
1.1 Latar Belakang	1
1.2 Rumusan Masalah	2
1.3 Batasan Masalah	3
1.4 Tujuan	3
1.5 Manfaat	3
1.6 Sistematika Penulisan	4
BAB II DASAR TEORI	6
2.1 Permainan	6
2.2 Klasifikasi <i>Game</i>	7
2.3 Klasifikasi <i>Rating Game</i> Berdasarkan Usia	8
2.4 <i>Unified Modelling Language</i>	9
2.4.1 <i>Use Case Diagram</i>	10
2.4.2 Class Diagram	10
2.4.3 EER Diagram	10
2.5 MySQL	10
2.6 Framework	11
2.6.1 <i>Design Patern: MVC (Model, View, Controller)</i>	12
2.6.2 Framework Code Igniter	13
2.7 Pengujian Perangkat Lunak	13
2.7.1 Teknik Pengujian	14

2.7.2 Strategi Pengujian.....	16
BAB III METODOLOGI DAN PERANCANGAN	19
3.1 Metodologi.....	19
3.1.1 Studi Literatur.....	20
3.1.2 Perancangan.....	20
3.1.3 Implementasi	22
3.1.4 Pengujian	23
3.2 Perancangan.....	24
3.2.1 Game Concept Design	24
3.2.2 Technical Design	32
BAB IV IMPLEMENTASI	52
4.1 Spesifikasi Sistem.....	52
4.1.1 Spesifikasi Lingkungan Perangkat Keras	52
4.1.2 Spesifikasi Lingkungan Perangkat Lunak	53
4.2 Batasan-Batasan Dalam Implementasi	53
4.3 Implementasi Algoritma <i>Gameplay</i>	54
4.3.1 Function	54
4.3.2 Procedure	73
4.3.3 Event.....	176
4.4 Implementasi <i>Web</i>	180
4.4.1 Implementasi Kelas <i>Model</i>	180
4.4.2 Implementasi Kelas <i>Controller</i>	193
4.4.3 Implementasi <i>View</i>	212
BAB V PENGUJIAN DAN ANALISIS.....	214
5.1 Pengujian Unit	214
5.1.1 Pengujian Unit Pada <i>Function</i> Dalam Perangkat Lunak	214
5.1.2 Analisis pengujian Unit	230
5.2 Pengujian Integrasi	230
5.2.1 Pengujian Integrasi Pada <i>Function</i> dan <i>Procedure</i> Dalam Perangkat Lunak	231

5.2.2 Analisis Pengujian Integrasi	264
5.3 Pengujian Validasi.....	264
5.3.1 Daftar kebutuhan fungsional dan non-fungsional	264
5.3.2 Kasus uji validasi.....	271
5.3.3 Hasil Pengujian Validasi	283
5.3.4 Analisis Pengujian Validasi.....	284
BAB VI KESIMPULAN DAN SARAN	286
6.1 Kesimpulan.....	286
6.2 Saran	286
DAFTAR PUSTAKA	DP-1



DAFTAR GAMBAR

Gambar 2.1 Flow graph [PRE-01]	15
Gambar 3.1 Diagram Alir Pengerjaan Tugas Akhir.....	19
Gambar 3.2 Diagram Alir Proses Perancangan.....	21
Gambar 3.3 Diagram Alir Proses Implementasi	22
Gambar 3.4 Diagram Alir Proses Pengujian	23
Gambar 3.5 Use Case Pendaftaran Pemain.....	33
Gambar 3.6 Use Case Masuk Dunia Permainan	34
Gambar 3.7 Use Case Pengumpulan Sumber Daya	35
Gambar 3.8 Use Case Pembangunan Bangunan	36
Gambar 3.9 Use Case Pelatihan Pasukan.....	38
Gambar 3.10 Use Case Kontak dengan Desa Lain	39
Gambar 3.11 Use Case Permainan Strategy Kerajaan Majapahit.....	42
Gambar 3.12 Class Diagram Model.....	43
Gambar 3.13 Class Diagram Controller.....	44
Gambar 3.14 EER Diagram Variables	45
Gambar 3.15 EER Diagram Attributes	46
Gambar 3.16 EER Diagram Players	46
Gambar 3.17 EER Diagram Villages	47
Gambar 3.18 EER Diagram Resources.....	48
Gambar 3.19 EER Diagram Buildings.....	49
Gambar 3.20 EER Diagram Troops	50
Gambar 3.21 EER Diagram Views	51
Gambar 4.1 Halaman Web Home	212
Gambar 5.1 Flow graph Pengujian Function get_attribute_id	215
Gambar 5.2 Flow graph Pengujian Function get_datetime_variables_from_id.	217
Gambar 5.3 Flow graph Pengujian Function get_troop_id	218
Gambar 5.4 Flow graph Pengujian Function get_new_attack.....	220
Gambar 5.5 Flow graph Pengujian Function get_new_idv	222
Gambar 5.6 Flow graph Pengujian Function get_new_idvas.....	225
Gambar 5.7 Flow graph Pengujian Function get_new_idvt.....	227
Gambar 5.8 Flow graph Pengujian Function get_varchar_variables_id	229
Gambar 5.9 Flow graph Pengujian Function get_village_axis	232
Gambar 5.10 Flow graph Pengujian Procedure get_village_location.....	235
Gambar 5.11 Flow graph Pengujian Function get_distance.....	239
Gambar 5.12 Flow graph Pengujian Function get_travel_time.....	244
Gambar 5.13 Flow graph Pengujian Function generate_village_resource_schedule_from_resources_buildings	248

Gambar 5.14 *Flow graph* Pengujian *Procedure order_attack* 259



DAFTAR TABEL

Tabel 3.1 Implementasi beserta jenis bahasa pemrograman dan perangkat lunak yang dipakai	22
Tabel 3.2 Main Game Concepts	24
Tabel 3.3 Kebutuhan Teknologi	25
Tabel 3.4 Sumber Daya Awal	28
Tabel 3. 5 Bangunan Penghasil Sumber Daya	29
Tabel 3. 6 Bangunan Penghasil Sumber Daya	29
Tabel 3.7 Bangunan Non Penghasil Sumber Daya	30
Tabel 3.8 Biaya & Waktu Pelatihan Pasukan	31
Tabel 3.9 Atribut Pasukan	31
Tabel 4.1 Spesifikasi Lingkungan Perangkat Keras Laptop	52
Tabel 4.2 Spesifikasi Lingkungan Perangkat Lunak Pengkodean	53
Tabel 4.3 Spesifikasi Lingkungan Perangkat Lunak Server	53
Tabel 4.4 Implementasi Function get_attribute_id	54
Tabel 4.5 Implementasi Function get_building_id	55
Tabel 4.6 Pseudocode Function get_datetime_variables	56
Tabel 4.7 Pseudocode Function get_datetime_variables_from_id	57
Tabel 4.8 Pseudocode Function get_datetime_variables_id	58
Tabel 4.9 Pseudocode Function get_distance	59
Tabel 4.10 Pseudocode Function get_integer_variables	60
Tabel 4.11 Pseudocode Function get_integer_variables_from_id	61
Tabel 4.12 Pseudocode Function get_integer_variables_id	62
Tabel 4.13 Pseudocode Function get_new_attack	62
Tabel 4.14 Pseudocode Function get_new_idv	64
Tabel 4.15 Pseudocode Function get_new_idvas	65
Tabel 4.16 Pseudocode Function get_new_idvt	66
Tabel 4.17 Pseudocode Function get_travel_time	67
Tabel 4.18 Pseudocode Function get_troop_id	68
Tabel 4.19 Pseudocode Function get_varchar_variables	69
Tabel 4.20 Pseudocode Function get_varchar_variables_from_id	70
Tabel 4.21 Pseudocode Function get_varchar_variables_id	71
Tabel 4.22 Pseudocode Function get_village_axis	72
Tabel 4.23 Pseudocode Function random_integer	72
Tabel 4.24 Pseudocode Procedure build_village	73
Tabel 4.25 Pseudocode Procedure buy_building	74
Tabel 4.26 Pseudocode Procedure buy_troop	76
Tabel 4.27 Pseudocode Procedure default_village	79

Tabel 4.28 Pseudocode Procedure default_village_buildings	80
Tabel 4.29 Pseudocode Procedure default_village_resources	81
Tabel 4.30 Pseudocode Procedure generate_village_resource_schedule_from_resources_buildings	82
Tabel 4.31 Pseudocode Procedure get_village_location.....	84
Tabel 4.32 Pseudocode Procedure insert_building_build_time.....	85
Tabel 4.33 Pseudocode Procedure insert_building_price	86
Tabel 4.34 Pseudocode Procedure insert_resource_building.....	87
Tabel 4.35 Pseudocode Procedure insert_troop_build_time.....	88
Tabel 4.36 Pseudocode Procedure insert_troop_price	89
Tabel 4.37 Pseudocode Procedure insert_troop_stat	90
Tabel 4.38 Pseudocode Procedure insert_village_location.....	92
Tabel 4.39 Pseudocode Procedure insert_village_resource	93
Tabel 4.40 Pseudocode Procedure insert_village_resource_schedule	94
Tabel 4.41 Pseudocode Procedure insert_village_troop	95
Tabel 4.42 Pseudocode Procedure manual_processing_event_and_scheduling..	97
Tabel 4.43 Pseudocode Procedure order_attack	97
Tabel 4.44 Pseudocode Procedure order_attack_simulation	101
Tabel 4.45 Pseudocode Procedure order_building.....	105
Tabel 4.46 Pseudocode Procedure order_troop	107
Tabel 4.47 Pseudocode Procedure player_registration	109
Tabel 4.48 Pseudocode Procedure random_village_axis.....	110
Tabel 4.49 Pseudocode Procedure reset_game_server	112
Tabel 4.50 Pseudocode Procedure sync_all_mem_variables.....	113
Tabel 4.51 Pseudocode Procedure sync_datetime_variables.....	114
Tabel 4.52 Pseudocode Procedure sync_integer_variables	115
Tabel 4.53 Pseudocode Procedure sync_varchar_variables.....	117
Tabel 4.54 Pseudocode Procedure village_building_registration.....	119
Tabel 4.55 Pseudocode Procedure village_registration	120
Tabel 4.56 Pseudocode Procedure villages_buildings_builds_scheduling.....	121
Tabel 4.57 Pseudocode Procedure villages_resources_scheduling	123
Tabel 4.58 Pseudocode Procedure villages_troops_attacks_scheduling	125
Tabel 4.59 Pseudocode Procedure villages_troops_builds_scheduling.....	142
Tabel 4.60 Pseudocode Procedure wars_loots_scheduling.....	145
Tabel 4.61 Pseudocode Procedure web_building_prices.....	147
Tabel 4.62 Pseudocode Procedure web_count_enemy_attack_schedules	147
Tabel 4.63 Pseudocode Procedure web_count_troop_attack_schedules	148
Tabel 4.64 Pseudocode Procedure web_count_troop_back_schedules	149
Tabel 4.65 Pseudocode Procedure web_count_village_building_build_schedules	150

Tabel 4.66 Pseudocode Procedure web_count_village_troop_build_schedules.	151
Tabel 4.67 Pseudocode Procedure web_count_war_loot_schedules	152
Tabel 4.68 Pseudocode Procedure web_default_active_village	153
Tabel 4.69 Pseudocode Procedure web_enemy_attack_schedules	154
Tabel 4.70 Pseudocode Procedure web_log_in	155
Tabel 4.71 Pseudocode Procedure web_order_attack.....	156
Tabel 4.72 Pseudocode Procedure web_order_building.....	157
Tabel 4.73 Pseudocode Procedure web_order_troop.....	158
Tabel 4.74 Pseudocode Procedure web_sign_up	159
Tabel 4.75 Pseudocode Procedure web_troop_attack_schedules	160
Tabel 4.76 Pseudocode Procedure web_troop_back_schedules	161
Tabel 4.77 Pseudocode Procedure web_troop_prices.....	162
Tabel 4.78 Pseudocode Procedure web_troop_stats	163
Tabel 4.79 Pseudocode Procedure web_username_availability	164
Tabel 4.80 Pseudocode Procedure web_village_building_amounts.....	165
Tabel 4.81 Pseudocode Procedure web_village_building_build_schedules.....	166
Tabel 4.82 Pseudocode Procedure web_village_buildings.....	167
Tabel 4.83 Pseudocode Procedure web_village_location.....	167
Tabel 4.84 Pseudocode Procedure web_village_ranking.....	169
Tabel 4.85 Pseudocode Procedure web_village_resource_schedules.....	170
Tabel 4.86 Pseudocode Procedure web_village_resources.....	171
Tabel 4.87 Pseudocode Procedure web_village_troop_build_schedules.....	172
Tabel 4. 88 Pseudocode Procedure web_villages_locations.....	173
Tabel 4.89 Pseudocode Procedure web_villages_rankings	174
Tabel 4.90 Pseudocode Procedure web_villages_troops	175
Tabel 4.91 Pseudocode Procedure web_war_loot_schedules.....	176
Tabel 4.92 Pseudocode Event event_sync_all_mem_variables.....	177
Tabel 4.93 Pseudocode Event event_villages_buildings_builds_scheduling	177
Tabel 4.94 Pseudocode Event event_villages_resources_scheduling.....	178
Tabel 4.95 Pseudocode Event event_villages_troops_attacks_scheduling.....	178
Tabel 4.96 Pseudocode Event event_villages_troops_builds_scheduling	179
Tabel 4.97 Pseudocode Event event_wars_loots_scheduling.....	180
Tabel 4. 98 Pseudocode Class Buildings_model	180
Tabel 4. 99 Pseudocode Class Form_validation_model	182
Tabel 4. 100 Pseudocode Class Log_in_model	184
Tabel 4.101 Pseudocode Class Resources_model	185
Tabel 4.102 Pseudocode Class Sign_up_model.....	187
Tabel 4.103 Pseudocode Class Troops_model.....	188
Tabel 4.104 Pseudocode Class Villages_model.....	191
Tabel 4.105 Pseudocode Class Buildings	193

Tabel 4.106 Pseudocode Class Form_validation	196
Tabel 4.107 Pseudocode Class Home	198
Tabel 4.108 Pseudocode Class Log_in	199
Tabel 4.109 Pseudocode Class Resources	200
Tabel 4.110 Pseudocode Class Sign_up.....	203
Tabel 4.111 Pseudocode Class Troops.....	204
Tabel 4.112 Pseudocode Class Villages.....	209
Tabel 5.1 Pemodelan <i>Flow Graph</i> Algoritma <i>Function</i> get_attribute_id	214
Tabel 5.2 Kasus Uji pada Pengujian <i>Function</i> get_attribute_id	216
Tabel 5.3 Pemodelan <i>Flow Graph</i> Algoritma <i>Function</i> get_datetime_ variables_from_id	216
Tabel 5.4 Kasus Uji pada Pengujian <i>Function</i> get_datetime_variables_from_id	217
Tabel 5.5 Pemodelan <i>Flow Graph</i> Algoritma <i>Function</i> get_troop_id	218
Tabel 5.6 Kasus Uji pada Pengujian <i>Function</i> get_troop_id	219
Tabel 5.7 Pemodelan <i>Flow Graph</i> Algoritma <i>Function</i> get_new_attack	219
Tabel 5.8 Kasus Uji pada Pengujian <i>Function</i> get_new_attack.....	221
Tabel 5.9 Pemodelan <i>Flow Graph</i> Algoritma <i>Function</i> get_new_idv.....	221
Tabel 5.10 Kasus Uji pada Pengujian <i>Function</i> get_new_idv.....	223
Tabel 5.11 Pemodelan <i>Flow Graph</i> Algoritma <i>Function</i> get_new_idvas	224
Tabel 5.12 Kasus Uji untuk Pengujian <i>Function</i> get_new_idvas	225
Tabel 5.13 Pemodelan <i>Flow Graph</i> Algoritma <i>Function</i> get_new_idvt	226
Tabel 5.14 Kasus Uji untuk Pengujian <i>Function</i> get_new_idvt	228
Tabel 5.15 Pemodelan <i>Flow Graph</i> Algoritma <i>Function</i> get_varchar_variables_id	228
Tabel 5.16 Kasus Uji pada Pengujian <i>Function</i> get_varchar_variables_id	230
Tabel 5.17 Pemodelan <i>Flow Graph</i> Algoritma <i>Function</i> get_village_axis	231
Tabel 5.18 Kasus Uji pada Pengujian <i>Function</i> get_village_axis.....	233
Tabel 5.19 Pemodelan <i>Flow Graph</i> Algoritma <i>Procedure</i> get_village_location	233
Tabel 5.20 Kasus Uji pada Pengujian <i>Procedure</i> get_village_location.....	236
Tabel 5.21 Pemodelan <i>Flow Graph</i> Algoritma <i>Function</i> get_distance	236
Tabel 5.22 Kasus Uji pada Pengujian <i>function</i> get_distance	240
Tabel 5.23 Pemodelan <i>Flow Graph</i> Algoritma <i>Function</i> get_travel_time	240
Tabel 5.24 Kasus Uji pada Pengujian <i>function</i> get_travel_time	245
Tabel 5.25 Pemodelan <i>Flow Graph</i> Algoritma <i>Function</i> generate_village_ resource_schedule_from_resources_buildings	245
Tabel 5.26 Kasus Uji pada Pengujian <i>Procedure</i> generate_village_resource_ schedule_from_resources_buildings	249
Tabel 5.27 Pemodelan <i>Flow Graph</i> Algoritma <i>Procedure</i> order_attack.....	250
Tabel 5.28 Kasus Uji pada Pengujian <i>Procedure</i> order_attack	261

Tabel 5.29 Daftar Berbagai Macam Kebutuhan Fungsional dan Non-Fungsional dari <i>Game Server Online</i> Permainan <i>Strategy</i> Kerajaan Majapahit.....	265
Tabel 5.30 Kasus Uji Validasi pada Pengujian Validasi <i>Game Server Online</i> Permainan <i>Strategy</i> Kerajaan Majapahit.....	271
Tabel 5.31 Hasil Uji Validasi pada Pengujian Validasi <i>Game Server Online</i> Permainan <i>Strategy</i> Kerajaan Majapahit.....	283



BAB I PENDAHULUAN

1.1 Latar Belakang

Indonesia pada hakikatnya ialah sebuah negeri yang kaya akan keanekaragaman sejarahnya. Mulai dari sejarah berdiri, berkembang serta runtuhnya berbagai kerajaan di tanah air hingga sejarah perjuangan pra kemerdekaan maupun pasca kemerdekaan. Salah satu bagian dari sejarah Indonesia yang cukup dikenal ialah Kerajaan Majapahit.

Majapahit, nama kerajaan besar yang pernah berhasil menyatukan serta berjaya di bumi nusantara. Puncak kejayaan diraih dibawah pimpinan Tribhuwana Tungga Dewi dengan mahapatihnya, yakni Mahapatih Gajahmada yang terkenal akan sumpah palapanya. Sejarah Majapahit secara khusus serta sejarah Indonesia secara umum merupakan harta tak ternilai warisan leluhur bangsa yang harus selalu dilestarikan.

Generasi muda sebagai generasi penerus bangsalah yang nantinya berperan dalam pelestarian sejarah di Indonesia. Namun, pengaruh budaya asing yang semakin gencar membuat generasi muda kurang tertarik untuk mempelajari budaya tradisional, bahkan sedikit demi sedikit mulai menjauhi dan melupakan budaya tradisional bangsa [WIJ-12:01]. Ironisnya, selain sektor kebudayaan, masalah tersebut berdampak pula pada sektor-sektor lain dan sektor sejarah termasuk di dalamnya.

Berkurangnya minat generasi muda terhadap sejarah bangsa salah satunya dikarenakan kurangnya media pembelajaran sejarah yang dirasa cukup menarik bagi mereka. Hingga saat ini, pembelajaran sejarah di Indonesia kebanyakan masih memakai metode narasi konvensional dengan buku sebagai media penyampaian materi. Materi narasi yang dipaparkanpun cenderung berupa tulisan.

Pada dasarnya otak akan lebih tertarik pada warna dan bentuk daripada hanya berupa tulisan. Oleh karenanya, guna menimbulkan ketertarikan pada pelajaran sejarah maka perlu diberikan pendekatan dari sisi yang sekiranya mereka akan tertarik dengan itu [SAP-11:01]. *Game* dapat digunakan sebagai alternatif media pembelajaran sejarah, atau minimal bisa memancing minat generasi muda untuk lebih mendalami suatu materi sejarah dengan cara mengangkat tema sejarah pada *game* yang dibuat.

Pembuatan aplikasi *game* kebanyakan hanya dapat dimainkan oleh satu pemain saja (*singleplayer*), seperti *game* Hangman, Sudoku, BrickBreaker, dan lain sebagainya, yang mungkin dapat menyebabkan kejenuhan pemain, karena hanya berinteraksi dengan komputer (*artificial intelligence*) [WEN-12:02]. Oleh karenanya sebuah aplikasi *game multiplayer* yang memungkinkan adanya interaksi antar pemain dapat menjadi solusi untuk mengatasi kejenuhan tatkala berinteraksi dengan *artificial intelligence* saja.

Game multiplayer bisa dirancang sebagai suatu aplikasi *client-server* dengan interaksi antar pemain yang terkoneksi secara *online* menggunakan media *internet* yang penggunaanya bisa dibilang masih cukup tinggi di Indonesia. Kementerian Komunikasi dan Informatika (Kemenkominfo) mengungkapkan pengguna internet di Indonesia saat ini mencapai 63 juta orang [AN4-13]. Mengingat kondisi tersebut beserta berbagai permasalahan yang telah dipaparkan sebelumnya, penulis mengusulkan sebuah *Game Server Online* pada Permainan *Strategy* Kerajaan Majapahit. *Game Server* tersebut ialah *game server* berbasis *web* yang nantinya akan diintegrasikan dengan suatu aplikasi *Game Client* berbasis *android*.

1.2 Rumusan Masalah

Berdasarkan pada permasalahan yang telah dipaparkan pada bagian latar belakang, maka dapat disusun beberapa rumusan masalah dalam tugas akhir ini yakni sebagai berikut :

1. Bagaimana merancang *gameplay* dan desain aplikasi perangkat lunak dari *game server online* permainan *strategy* Kerajaan Majapahit ?

2. Bagaimana membangun aplikasi perangkat lunak *game server online* dari desain aplikasi perangkat lunak yang telah dirancang ?
3. Bagaimana menguji aplikasi perangkat lunak *game server online* yang telah dibuat ?

1.3 Batasan Masalah

Sesuai dengan rumusan masalah yang telah dipaparkan, agar pembahasan bisa lebih terfokus, maka batasan masalah yang perlu diberlakukan dalam tugas akhir ini ialah sebagai berikut :

1. *Game server online* yang dirancang serta dibangun merupakan *game server* berbasis *web*.
2. Implementasi rancangan *game server online* menggunakan bahasa pemrograman PHP dan bahasa *query* MySQL.
3. Algoritma *gameplay* dari permainan diimplementasikan sepenuhnya dengan bahasa *query* pada MySQL.
4. Masukan data pada *HTTP request* ke *game server online* menggunakan metode POST.
5. Keluaran data atau *response* dari *game server online* dalam format JSON.
6. Rancang bangun *game server online* dengan dibantu MVC *framework Code Igniter*.

1.4 Tujuan

Tujuan penulisan tugas akhir ini ialah merancang, membangun dan menguji aplikasi perangkat lunak *game server online* permainan *strategy* Kerajaan Majapahit yang dapat terintegrasi dengan berbagai jenis *compatible game client* secara umum dan *compatible game client* berbasis *android* secara khusus.

1.5 Manfaat

Penelitian ini diharapkan dapat bermanfaat untuk berbagai pihak. Manfaat dari penelitian ini adalah sebagai berikut:

1. Bagi penulis :

- a. Mengaplikasikan ilmu yang didapat selama mengikuti perkuliahan di Teknik Informatika Universitas Brawijaya.
 - b. Mendapatkan pengetahuan tentang proses pembangunan aplikasi perangkat lunak *game server online* berbasis *web* yang dapat terintegrasi dengan *game client* berbasis *android*.
2. Bagi pengguna :
- a. Sebagai media pembelajaran alternatif yang dapat dipakai untuk mempelajari sejarah Kerajaan Majapahit.
 - b. Sebagai sarana hiburan yang memiliki unsur edukasi sejarah di dalamnya.

1.6 Sistematika Penulisan

Sistematika penulisan laporan tugas akhir ini sebagai berikut:

Bab I Pendahuluan

Pendahuluan berisi tentang latar belakang, rumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian, dan sistematika penulisan.

Bab II Landasan Teori

Landasan teori berisi tentang dasar teori secara luas mengenai berbagai macam aplikasi perangkat lunak dan metode yang diperlukan untuk rancang bangun *game server online* pada permainan *strategy* Kerajaan Majapahit.

Bab III Metodologi dan Perancangan

Metodologi dan perancangan berisi tentang langkah-langkah dalam merancang *gameplay* serta desain aplikasi perangkat lunak pada *game server online* permainan *strategy* Kerajaan Majapahit yang berbasis *web* dan dapat terintegrasi dengan berbagai *compatible game client*. Pada bagian metodologi akan dijelaskan langkah-langkah yang akan dilakukan yakni perumusan masalah, studi literatur, perancangan permainan, implementasi, dan pengujian. Kesimpulan dan saran disertakan sebagai catatan atas aplikasi perangkat lunak dan

kemungkinan arah pengembangan aplikasi perangkat lunak selanjutnya. Pada bagian perancangan berisi tentang perencanaan *game server online* yang dikembangkan, dengan membuat perancangan *gameplay* dan perancangan teknis.

Bab IV Implementasi

Implementasi berisi tentang pembangunan aplikasi perangkat lunak *game server online* berbasis *web* dari perancangan yang telah dibuat, meliputi implementasi algoritma *gameplay* dengan bahasa *query* pada MySQL serta penggunaan MVC *framework Code Igniter*.

Bab V Pengujian dan Analisis

Pengujian dan analisis berisi mengenai pengujian aplikasi serta analisa hasil perancangan aplikasi, hasil analisa output aplikasi, dan pembahasan terjadinya malfungsi atau *bug* (apabila ditemukan malfungsi).

Bab VI Penutup

Penutup berisi kesimpulan yang diambil berdasarkan analisa dari proses pengembangan yang meliputi keunikan, kelebihan dan kekurangan, serta berbagai saran untuk penyempurnaan atau pengembangan lebih lanjut dari aplikasi perangkat lunak *game server online* yang telah dibuat.

BAB II

DASAR TEORI

2.1 Permainan

Permainan, atau dalam bahasa Inggris disebut dengan *game* adalah aktivitas yang dimainkan untuk olahraga atau hiburan menurut aturan [LXP-96]. Menurut Greg Costikyan, *game* adalah sebuah struktur interaktif dari nilai yang bersifat endogen dan membutuhkan pemain untuk berjuang mencapai tujuan [SCH-08:31]. “Sebuah struktur interaktif” memberi gambaran yang jelas bahwa dalam *game* pemain bermain dengan aktif, dan tidak pasif, dan pemain tersebut berinteraksi satu dengan lainnya.

Game memiliki struktur (yang didefinisikan dengan aturan-aturan) sehingga pemain dapat berinteraksi dan bagaimana pemain dapat berinteraksi. Endogen memiliki definisi “disebabkan oleh faktor di dalam organisme atau sistem” atau “dihasilkan dari dalam”. Jadi nilai yang bersifat endogen berarti nilai yang ada di dalam *game* hanya mempunyai nilai di dalam *game*. Sebagai contoh, uang pada *game* monopoli hanya memiliki nilai jika dimainkan pada *game* monopoli. Kata “berjuang” memiliki arti bahwa di dalam *game* terdapat konflik yang harus diselesaikan dalam rangka mencapai tujuan.

Menurut pendapat yang lain, yaitu Tracy Fullerton, Chris Swain, and Steven Hoffman, *game* adalah sebuah sistem formal dan tertutup yang mengikutsertakan pemain dalam konflik yang terstruktur dan terselesaikan dengan hasil yang tidak seimbang [SCH-08:33]. Sebagai sistem, *game* terbentuk dari berbagai elemen yang bekerja bersama. *Formal* adalah kata lain untuk menyatakan bahwa sistem secara jelas terdefinisi, dalam hal ini yaitu aturan-aturan. *Closed* berarti bahwa terdapat batasan dalam sistem. “Batasan” dalam hal ini dapat dikatakan sebagai *magical effect*.

Johan Huizinga mengatakannya “the magic circle” [SCH-08:34]. Sebagai contoh, jika kita “berada dalam game”, kita memiliki perasaan dan pemikiran yang berbeda dibanding jika kita “berada di luar game”. “Hasil yang tidak seimbang” memiliki arti bahwa dalam *game* selalu diperoleh hasil menang atau kalah.

2.2 Klasifikasi *Game*

Pengelompokan *game* dilakukan berdasarkan tindakan yang dilakukan dalam *game* tersebut, topik yang diangkat dalam *game* atau pemain *game* tersebut. *Game* atau permainan juga mungkin memiliki lebih dari satu *genre*. Berikut adalah penjelasan mengenai beberapa kategori *genre game* [REP-08]:

1. *Action game*: *game* yang menuntut pemainnya terus melakukan gerakan seperti bergerak, menyerang, bergerak, bereaksi, lalu bergerak lagi pada karakter permainannya. Pada *game* ini lebih diutamakan aksinya daripada jalan ceritanya.
2. *Adventure game*: *game* petualangan dengan teka teki muncul selama perjalanannya.
3. *Casual game*: berisi permainan seperti catur, kartu atau biasa juga *gameshow* yang dikemas dalam *game*.
4. *Educational game*: *game* yang menekankan pada pembelajaran
5. *Role-playing games (RPG)*: *game* jenis ini memiliki dunia yang luas untuk dijelajahi. Grup dari pemainnya akan menjelajahi *terrain* untuk mencari objek yang diinginkan, dan meningkatkan *experience* serta status dengan mengalahkan monster dan hambatan selama perjalanannya.
6. *Simulations game*: *game* yang meniru realita di kehidupan sehari-hari
7. *Sports games*: *game* dengan tema olahraga. Saat ini dapat berupa *POV (point of view) game*, juga bisa berupa *strategy game* dan pemain biasanya memiliki peran sebagai manajer.

8. *Strategy games*: game yang membutuhkan pemikiran dan perencanaan. Pemenang game ditentukan dengan “*battle of the minds*”

2.3 Klasifikasi *Rating Game* Berdasarkan Usia

Entertainment Software Rating Board (ESRB) ratings menyediakan informasi tentang konten atau isi dari sebuah *game*. Tabel 2.1 menunjukkan pengklasifikasian *rating game* berdasarkan usia yang disesuaikan dengan konten sebuah *game*.

Tabel 2.1 Klasifikasi game berdasarkan usia [ESR-12]

Logo	Keterangan
	EC (Early Childhood): Klasifikasi <i>Game</i> ini memiliki konten yang mungkin cocok untuk anak usia 3 keatas. Dan tidak memiliki konten yang dianggap tidak pantas dilihat oleh anak dengan usia tersebut.
	E (Everyone): Klasifikasi <i>Game</i> ini memiliki konten yang mungkin cocok untuk anak dengan usia 6 atau lebih tua. Judul dalam kategori ini mungkin berisi kartun, fantasi atau kekerasan ringan dan / atau menggunakan bahasa ringan (tidak kasar).
	E10+ (Everyone 10 and older): Klasifikasi <i>Game</i> ini memiliki konten yang mungkin cocok untuk usia 10 atau lebih tua. Judul dalam kategori ini mungkin berisi kartun, fantasi atau kekerasan ringan, bahasa ringan dan / atau dengan tema sugestif minimal.
	T (Teen): Klasifikasi <i>Game</i> ini memiliki konten yang mungkin cocok untuk usia 13 atau lebih tua. Judul dalam kategori <i>game</i> ini mungkin berisi kekerasan, tema sugestif, humor kasar, minimal darah, simulasi perjudian, dan / atau menggunakan bahasa kuat yang minim.

	M (<i>Mature</i>): Klasifikasi <i>Game</i> ini memiliki konten yang mungkin cocok untuk orang usia 17 atau lebih tua. Judul dalam kategori ini mungkin berisi kekerasan intens, darah dan , konten seksual dan bahasa kuat.
	AO (<i>Adults Only</i>): Klasifikasi <i>Game</i> ini memiliki konten yang hanya dan harus dimainkan oleh orang 18 tahun keatas. Judul dalam kategori ini mungkin termasuk adegan kekerasan berkepanjangan intens dan konten seksual grafis dengan kontent yang bersisi gambar atau video telanjang.
	RP (<i>RATING PENDING</i>): Judul terdaftar sebagai RP (Rating Pending) telah disampaikan kepada ESRB dan sedang menunggu penilaian akhir. (Simbol ini hanya akan muncul dalam iklan sebelum merilis <i>game</i>.)

2.4 *Game Server*

Game server adalah suatu *server* yang digunakan oleh klien atau *gamer* untuk menjalankan *game multiplayer* komputer melalui suatu jaringan. *Server* ini di-*host* baik secara lokal atau jarak jauh. *Server* yang di-*host* secara lokal dapat diakses oleh mereka yang berkoneksi di intranet lokal saja dan *server* yang di-*host* pada *internet* keseluruhannya dapat diakses dari seluruh dunia [SEP-13].

2.5 *Unified Modelling Language*

Unified Modelling Language (UML) adalah sebuah "bahasa" yang telah menjadi standar dalam industri untuk visualisasi, merancang dan mendokumentasikan sistem piranti lunak. UML menawarkan sebuah standar untuk merancang model sebuah sistem [DHA-06].

2.5.1 Use Case Diagram

Use case diagram menggambarkan fungsionalitas yang diharapkan dari sebuah sistem. Yang ditekankan adalah “apa” yang diperbuat sistem, dan bukan “bagaimana”. Sebuah *use case* merepresentasikan sebuah interaksi antara aktor dengan sistem. *Use case* merupakan sebuah pekerjaan tertentu, misalnya login ke sistem, meng-*create* sebuah daftar belanja, dan sebagainya. Seorang/sebuah aktor adalah sebuah entitas manusia atau mesin yang berinteraksi dengan sistem untuk melakukan pekerjaan-pekerjaan tertentu [DHA-06].

2.5.2 Class Diagram

Class adalah sebuah spesifikasi yang jika diinstansiasi akan menghasilkan sebuah objek dan merupakan inti dari pengembangan dan desain berorientasi objek. *Class* menggambarkan keadaan (atribut/properti) suatu sistem, sekaligus menawarkan layanan untuk memanipulasi keadaan tersebut (metoda/fungsi). *Class diagram* menggambarkan struktur dan deskripsi *class*, *package* dan objek beserta hubungan satu sama lain seperti *containment*, pewarisan, asosiasi, dan lain-lain [DHA-06].

2.5.3 EER Diagram

Diagram E-R atau sering disebutkan secara panjang yaitu Diagram Entitas Relasional atau juga dalam versi bahasa inggris nya adalah *Entity Relationship Diagram* (ERD) merupakan bentuk perancangan dari tahap pembuatan *database* (basis data) [AN3-09]. Sedangkan Enhanced Entity Relationship (EER) ialah perpaduan antara Entiy Relationship (ER) dengan generalization, spesialisation dan categorization.

2.6 MySQL

MySQL merupakan *Database Management System* (DBMS) *tools open source* yang mendukung *multiuser*, *multithreaded*, populer dan *free*. MySQL

dimiliki oleh perusahaan asal Swedia, MySQL AB, di mana perusahaan tersebut memegang seluruh hak cipta atas kode sumbernya. MySQL AB didirikan oleh David Axmark, Allan Larsson, dan Michael “Monty” Widenius [AN1-12].

MySQL sebenarnya merupakan turunan salah satu konsep utama dalam *database* sejak lama, yaitu SQL (*Structured Query Language*). SQL adalah sebuah konsep pengoperasian *database*, terutama untuk pemilihan atau seleksi dan pemasukan data, yang memungkinkan pengoperasian data dikerjakan dengan mudah secara otomatis. Keandalan suatu sistem *database* (DBMS) dapat diketahui dari cara kerja *optimizer*-nya dalam melakukan proses perintah-perintah SQL, yang dibuat oleh user maupun program-program aplikasinya. Sebagai *database server*, MySQL dapat dikatakan lebih unggul dibandingkan *database server* lainnya dalam *query data* [AN1-12].

2.7 Framework

Framework secara sederhana dapat diartikan kumpulan dari fungsi-fungsi/prosedur-prosedur dan *class-class* untuk tujuan tertentu yang sudah siap digunakan sehingga bisa lebih mempermudah dan mempercepat pekerjaan seorang *programer*, tanpa harus membuat fungsi atau *class* dari awal [AN2-13]. Ada beberapa alasan mengapa menggunakan *Framework* [AN2-13] :

1. Mempercepat dan mempermudah pembangunan sebuah aplikasi *web*.
2. Relatif memudahkan dalam proses *maintenance* karena sudah ada pola tertentu dalam sebuah *framework* (dengan syarat *programmer* mengikuti pola standar yang ada).
3. Umumnya *framework* menyediakan fasilitas-fasilitas yang umum dipakai sehingga kita tidak perlu membangun dari awal (misalnya validasi, ORM, *pagination*, *multiple database*, *scaffolding*, pengaturan *session*, *error handling*, dll).
4. Lebih bebas dalam pengembangan jika dibandingkan CMS.

2.7.1 Design Patern: MVC (Model, View, Controller)

Model View Controller merupakan suatu konsep yang cukup populer dalam pembangunan aplikasi *web*, berawal pada bahasa pemrograman *Small Talk*, MVC memisahkan pengembangan aplikasi berdasarkan komponen utama yang membangun sebuah aplikasi seperti manipulasi data, *user interface*, dan bagian yang menjadi kontrol aplikasi [AN2-13]. Terdapat 3 jenis komponen yang membangun suatu MVC pattern dalam suatu aplikasi yaitu [AN2-13] :

1. *View*, merupakan bagian yang menangani *presentation logic*. Pada suatu aplikasi *web* bagian ini biasanya berupa *file template* HTML, yang diatur oleh *controller*. *View* berfungsi untuk menerima dan merepresentasikan data kepada *user*. Bagian ini tidak memiliki akses langsung terhadap bagian *model*.
2. *Model*, biasanya berhubungan langsung dengan *database* untuk memanipulasi data (*insert, update, delete, search*), menangani validasi dari bagian *controller*, namun tidak dapat berhubungan langsung dengan bagian *view*.
3. *Controller*, merupakan bagian yang mengatur hubungan antara bagian *model* dan bagian *view*, *controller* berfungsi untuk menerima *request* dan *data* dari *user* kemudian menentukan apa yang akan diproses oleh aplikasi.

Dengan menggunakan prinsip MVC suatu aplikasi dapat dikembangkan sesuai dengan kemampuan *developer*-nya, yaitu *programmer* yang menangani bagian *model* dan *controller*, sedangkan *designer* yang menangani bagian *view*, sehingga penggunaan arsitektur MVC dapat meningkatkan *maintanability* dan organisasi kode. Walaupun demikian dibutuhkan komunikasi yang baik antara *programmer* dan *designer* dalam menangani variabel-variabel yang akan ditampilkan [AN2-13]. Ada beberapa kelebihan *CodeIgniter* (CI) dibandingkan dengan *Framework* PHP lain [AN2-13]:

1. Performa sangat cepat : salah satu alasan tidak menggunakan framework adalah karena eksekusinya yang lebih lambat daripada PHP *from the scratch*,

- tapi *Codeigniter* sangat cepat bahkan mungkin bisa dibilang *codeigniter* merupakan *framework* yang paling cepat dibanding *framework* yang lain.
2. Konfigurasi yang sangat minim (*nearly zero configuration*) : tentu saja untuk menyesuaikan dengan database dan keleluasaan *routing* tetap diizinkan melakukan konfigurasi dengan mengubah beberapa file konfigurasi seperti *database.php* atau *autoload.php*, namun untuk menggunakan *codeigniter* dengan setting standard, anda hanya perlu merubah sedikit saja file pada folder *config*.
 3. Banyak komunitas: dengan banyaknya komunitas CI ini, memudahkan kita untuk berinteraksi dengan yang lain, baik itu bertanya atau teknologi terbaru.
 4. Dokumentasi yang sangat lengkap : Setiap paket instalasi *codeigniter* sudah disertai *user guide* yang sangat bagus dan lengkap untuk dijadikan permulaan, bahasanya pun mudah dipahami.
 5. Dan banyak lagi yang lainnya.

2.7.2 Framework Code Igniter

Code Igniter adalah aplikasi *open source* yang berupa *framework* dengan model MVC (*Model, View, Controller*) untuk membangun *website* dinamis dengan menggunakan PHP. *CodeIgniter* memudahkan developer untuk membuat aplikasi *web* dengan cepat dan mudah dibandingkan dengan membuatnya dari awal. *Code Igniter* dirilis pertama kali pada 28 Februari 2006 [AN2-13].

2.8 Pengujian Perangkat Lunak

Pengujian (*testing*) arsitektur dari perangkat lunak berorientasi objek menghasilkan sekumpulan *layered subsystems* yang mengenkapsulasi kelas-kelas yang berkolaborasi. Setiap elemen sistem (*subsistem* dan *class*) melakukan fungsi yang membantu untuk mencapai kebutuhan sistem. Hal ini sangat penting untuk menguji sebuah *object-oriented system* pada berbagai macam level yang berbeda dalam sebuah usaha untuk menemukan kesalahan-kesalahan yang mungkin terjadi

dari kolaborasi kelas-kelas dan komunikasi subsistem melewati *architetural layer* [PRE-01:631].

2.8.1 Teknik Pengujian

Pengujian perangkat lunak memerlukan perancangan kasus uji (*test case*) agar dapat menemukan kesalahan dalam waktu singkat dan usaha minimum. Berbagai macam metode perancangan kasus uji telah berevolusi. Metode-metode ini menyediakan *developer* pendekatan sistematis untuk pengujian. Terlebih lagi metode-metode ini menyediakan mekanisme yang dapat membantu memastikan kelengkapan dari pengujian dan menyediakan kemungkinan tertinggi untuk menemukan kesalahan-kesalahan dalam perangkat lunak [PRE-01]. Teknik atau metode perancangan kasus uji yang digunakan adalah *black-box testing* dan *white-box testing*.

2.8.1.1 White Box Testing

White-box testing atau *glass-box testing* merupakan sebuah metode perancangan kasus uji yang menggunakan struktur kontrol dari perancangan prosedural untuk memperoleh kasus uji [PRE-01]. Ada dua jenis pengujian yang termasuk *white-box testing* yaitu *basis path testing* dan *control structure testing*.

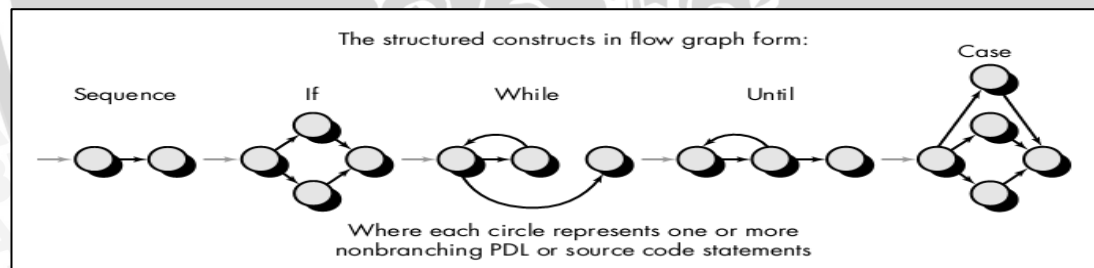
Pada tugas akhir ini menggunakan *basis path testing* yang diusulkan pertama kali oleh Tom McCabe [PRE-01]. *Basis path testing* ini memungkinkan perancang kasus uji memperoleh ukuran kompleksitas logis dari sebuah perancangan prosedural dan menggunakan pengukuran ini sebagai pedoman untuk mendefinisikan *basis set* dari jalur eksekusi (*execution path*). *Test case* yang dilakukan untuk menggunakan *basis set* tersebut dijamin untuk menggunakan setiap *statement* di dalam program paling tidak sekali selama pengujian. Sebelum metode *basis path* dapat diperkenalkan, notasi sederhana untuk representasi aliran kontrol yang disebut diagram alir (*flow graph*) harus diperkenalkan. Contoh flowgraph dapat dilihat pada Gambar 2.2.

Cyclomatic complexity adalah metrik perangkat lunak yang memberikan pengukuran kuantitatif terhadap kompleksitas logis suatu program. Bila metrik ini digunakan dalam konteks metode pengujian *basis path*, maka nilai yang terhitung untuk *cyclomatic complexity* menentukan jumlah jalur independen (*independent path*) dalam *basis set* suatu program dan memberi batas atas bagi jumlah pengujian yang harus dilakukan untuk memastikan bahwa semua *statement* telah dieksekusi sedikitnya satu kali.

Jalur independen adalah jalur yang melalui program yang mengenalkan sedikitnya satu rangkaian *statement* proses baru atau suatu kondisi baru.

1. Jumlah *region* pada *flow graph* sesuai dengan *cyclomatic complexity*.
2. *Cyclomatic complexity* $V(G)$, untuk grafik G adalah $V(G) = E - N + 2$. E adalah jumlah *edge*, dan N adalah jumlah *node*.

$V(G) = P + 1$. P adalah jumlah *predicate node* yaitu *node* yang merupakan kondisi (ada 2 atau lebih *edge* akan keluar *node* ini).



Gambar 2.1 Flow graph [PRE-01]

2.8.1.2 Black Box Testing

Pengujian *black box* atau *behavioral testing* berfokus pada persyaratan fungsional perangkat lunak [PRE-01]. Dengan demikian, pengujian *black-box* memungkinkan perekayasa perangkat lunak mendapatkan serangkaian kondisi *input* yang sepenuhnya menggunakan semua persyaratan fungsional untuk semua program. Pengujian *black-box* bukan merupakan alternatif dari teknik *white-box*,

tetapi merupakan pendekatan komplementer yang kemungkinan besar mampu mengungkap kelas kesalahan daripada metode *white-box*. Pengujian *black-box* berusaha menemukan kesalahan dalam kategori berikut :

1. Fungsi-fungsi yang tidak benar atau hilang
2. Kesalahan *interface*
3. Kesalahan dalam struktur data atau akses *database* eksternal
4. Kesalahan kinerja
5. Inisialisasi dan kesalahan terminasi.

2.8.2 Strategi Pengujian

Strategi untuk pengujian perangkat lunak mengintegrasikan metode desain *test case* perangkat lunak ke dalam sederetan langkah yang direncanakan dengan baik, dan hasilnya adalah konstruksi perangkat lunak yang berhasil [PRE-01]. Sejumlah strategi pengujian perangkat lunak telah diusulkan di dalam literatur. Strategi pengujian harus mengakomodasi pengujian tingkat rendah yang diperlukan untuk membuktikan bahwa segmen kode sumber yang kecil telah diimplementasikan dengan tepat, demikian juga pengujian tingkat tinggi yang memvalidasi fungsi-fungsi sistem mayor yang berlawanan dengan kebutuhan pelanggan. Proses pengujian dimulai dengan pengujian yang berfokus pada setiap modul secara individual (*unit testing*), dilanjutkan dengan pengujian integrasi (*integration testing*) dan berakhir pada pengujian validasi (*validation testing*) [PRE-01].

2.8.2.1 Pengujian Unit

Pengujian unit berfokus pada usaha verifikasi pada inti terkecil dari desain perangkat lunak, yakni modul. Dengan menggunakan gambaran desain prosedural sebagai panduan, jalur kontrol yang penting diuji untuk mengungkap kesalahan di dalam batas modul tersebut. Kompleksitas relatif dari pengujian dan

kesalahan yang diungkap dibatasi oleh ruang lingkup batasan yang dibangun untuk pengujian unit. Pengujian unit biasanya berorientasi pada *white-box*, dan langkahnya dapat dilakukan secara paralel untuk model bertingkat. *Interface* modul diuji untuk memastikan bahwa informasi secara tepat mengalir masuk dan keluar dari inti program yang diuji. Struktur data lokal diuji untuk memastikan bahwa data yang tersimpan secara temporal dapat tetap menjaga integritasnya selama semua langkah di dalam suatu algoritma dieksekusi. Kondisi batas diuji untuk memastikan bahwa modul beroperasi dengan tepat pada batas yang ditentukan untuk membatasi pemrosesan. Semua jalur independen (jalur dasar) yang melalui struktur kontrol dipakai sedikitnya satu kali. [PRE-01].

2.8.2.2 Pengujian Integrasi

Pengujian integrasi memastikan bahwa dua atau lebih unit atau integrasi lainnya dapat bekerja bersama dengan baik dan agar lebih fokus pada antarmuka yang terspesifikasi pada *low-level design*[SRJ-12].

2.8.2.3 Pengujian Validasi

Pada kulminasi pengujian terintegrasi, perangkat lunak secara lengkap dirakit sebagai suatu paket, kesalahan *interfacing* telah diungkap dan dikoreksi, dan seri akhir dari pengujian perangkat lunak, yaitu pengujian validasi dapat dimulai. Validasi dapat ditentukan dengan berbagai cara, tetapi definisi yang sederhana adalah bahwa validasi berhasil bila perangkat lunak berfungsi dengan cara yang dapat diharapkan secara bertanggung jawab oleh pelanggan. Validasi perangkat lunak dicapai melalui sederetan pengujian *black-box* yang memperlihatkan konformitas dengan persyaratan. Rencana pengujian menguraikan kelas-kelas pengujian yang akan dilakukan, dan prosedur pengujian menentukan *test case* spesifik yang akan digunakan untuk mengungkap kesalahan dalam konformitas dengan persyaratan. Baik rencana dan prosedur didesain untuk memastikan apakah semua persyaratan fungsional dipenuhi; semua persyaratan kinerja dicapai; dokumentasi benar dan direkayasa oleh manusia; dan persyaratan

lainnya dipenuhi (transportabilitas, kompatibilitas, pembetulan kesalahan, maintainabilitas) [PRE-01].

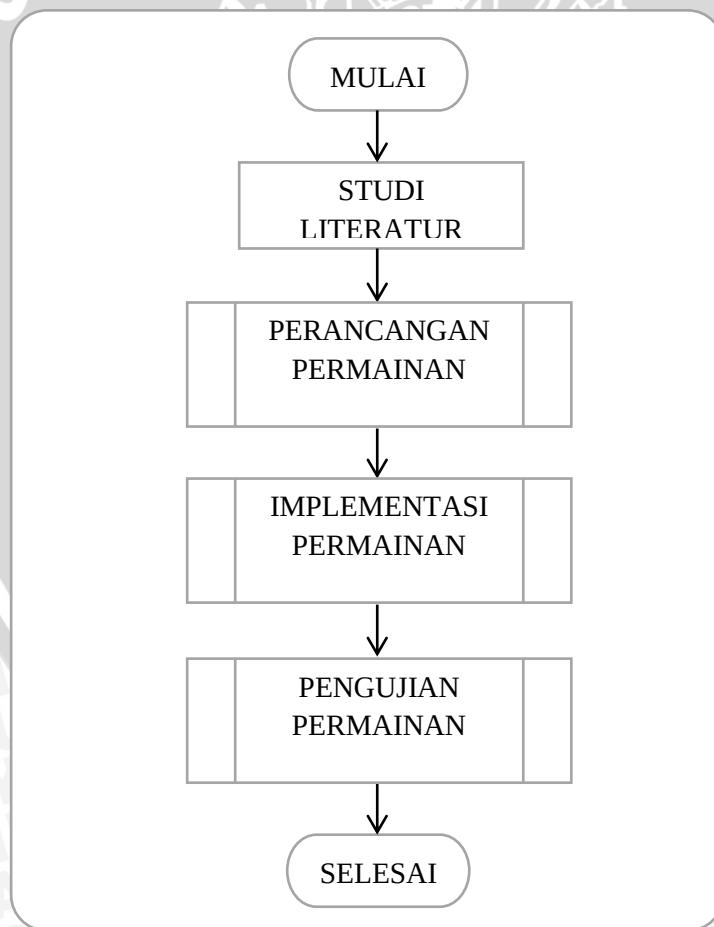


BAB III

METODOLOGI DAN PERANCANGAN

3.1 Metodologi

Pada bab ini dijelaskan langkah-langkah atau proses-proses yang akan dilakukan dalam pengerjaan tugas akhir ini, yakni studi literatur, penyusunan dasar teori, perancangan permainan, implementasi permainan serta pengujian pada aplikasi perangkat lunak yang akan dibuat. Kesimpulan dan saran disertakan sebagai catatan atas aplikasi dan kemungkinan arah pengembangan lebih lanjut dari aplikasi yang telah dibangun. Gambar 3.1 menunjukkan diagram alir pengerjaan tugas akhir.



Gambar 3.1 Diagram Alir Pengerjaan Tugas Akhir

3.1.1 Studi Literatur

Studi literatur ialah proses mempelajari serta memahami mengenai berbagai penjelasan atas dasar teori yang dibutuhkan untuk menunjang atau mendukung proses-proses dalam pengerjaan tugas akhir ini yang meliputi :

1. Perancangan permainan yang meliputi perancangan game dan perancangan desain perangkat lunak.
2. Implementasi game.
3. Pengujian game.
4. Penulisan skripsi.

Teori-teori pendukung tersebut diperoleh dari berbagai sumber yakni buku, jurnal, e-book, serta dokumentasi project. Dari berbagai sumber refrensi tersebut bisa didapatkan dan dibuat gambaran mengenai detil teknis maupun non-teknis yang diperlukan. Studi literatur yang dipakai ialah :

1. Dasar teori perancangan permainan.
2. Dasar teori perancangan perangkat lunak.
3. Dasar teori pemrograman dalam MySQL.
4. Dasar teori pemrograman PHP dengan *framework* Code Igniter.
5. Dasar teori penjelasan serta penggunaan JSON.
6. Dasar teori pengujian perangkat lunak.

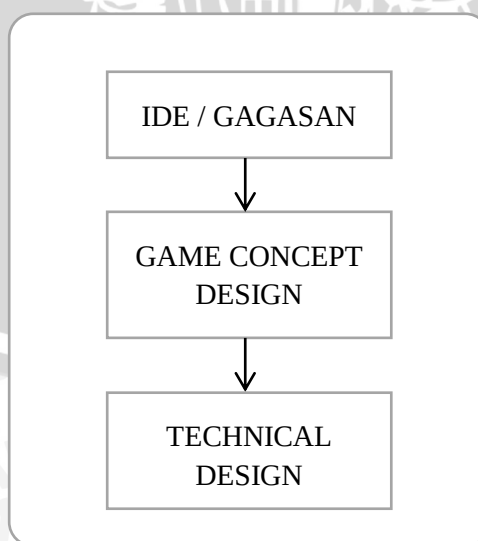
3.1.2 Perancangan

Perancangan merupakan proses merancang desain permainan yang meliputi desain *gameplay* serta desain perangkat lunak. Tujuan utama dari perancangan ialah memvisualisasikan secara konkrit, detil, terstruktur dan spesifik dari gagasan-gagasan bakal permainan yang akan dikembangkan dalam bentuk dokumen-dokumen perancangan yang berupa tulisan maupun gambar. Dokumen perancangan yang disusun nantinya akan dipakai sebagai gambaran serta pedoman arah implementasi perangkat lunak permainan yang akan dikembangkan.

Perancangan permainan dibagi menjadi 2 jenis yaitu *game concept design* serta *technical design*.

Game concept design dimulai dari mencari sebuah ide atau gagasan utama dari permainan. Dari gagasan utama yang telah ditemukan, dicari gagasan-gagasan pendukung guna menyusun *gameplay* permainan yakni berupa tema permainan, aturan atau *rules* permainan dan tujuan atau *goal* dalam permainan. Dari gagasan utama serta gagasan-gagasan pendukung yang telah terkumpul, maka dimulailah penyusunan dokumentasi berupa *Game Concept Design Document*. *Game Design Document* merupakan hasil dari *game concept design*.

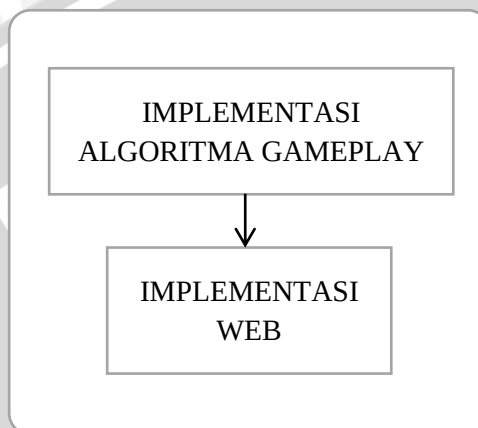
Langkah selanjutnya yang akan dilakukan ialah *technical design*. *Technical design* ialah proses perancangan dan penyusunan dokumen-dokumen teknis yang berhubungan langsung dengan implementasi perangkat lunak permainan. Dengan kata lain *technical design* merupakan proses yang menjembatani antara *game concept design* dengan implementasi perangkat lunak permainan. Penyusunan dokumen *technical design* mengacu atau berpedoman dari *Game Design Document* yang telah dibuat. Dokumen yang dihasilkan dalam *technical design* nantinya akan berupa UML (*Unified Modeling Language*). Gambar 3.2 menunjukkan diagram alir dalam proses perancangan.



Gambar 3.2 Diagram Alir Proses Perancangan

3.1.3 Implementasi

Proses implementasi perangkat lunak dilakukan dengan berpedoman atau mengacu kepada dokumen perancangan teknis yang telah disusun. Implementasi yang akan dilakukan ialah implementasi algoritma gameplay dan implementasi web. Gambar 3.3 menunjukkan diagram alir dalam proses implementasi.



Gambar 3.3 Diagram Alir Proses Implementasi

Tabel 3.1 menunjukkan implementasi yang dilakukan beserta jenis bahasa pemrograman dan perangkat lunak yang dipakai dalam pengerjaannya.

Tabel 3.1 Implementasi beserta jenis bahasa pemrograman dan perangkat lunak yang dipakai

Jenis Implementasi	Perangkat Lunak
Implementasi algoritma <i>gameplay</i>	MySQL dengan MySQL Workbench
Implementasi <i>web</i>	PHP dengan Notepad++

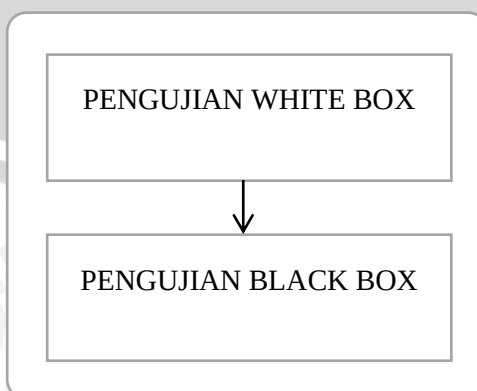
Implementasi algoritma *gameplay* dilakukan sepenuhnya dengan bahasa *query* pada MySQL. Fitur-fitur MySQL yang dipakai nantinya antara lain *table* mulai dari InnoDB *table*, Blackhole *table* hingga Memory *table*, *trigger*, *procedure*, *function* serta *event* sebagai pengeksekusi *procedure-procedure* terjadwal secara periodik. Hasil dari implementasi algoritma *gameplay* nantinya

ialah kumpulan *procedure* yang selanjutnya bakal dipakai serta diklasifikasikan sesuai dokumen perancangan teknis pada saat implementasi *web* dengan bantuan *framework* Code Igniter.

Pada implementasi *web* akan dikembangkan PHP based website yang berfungsi sebagai *interface* atau antarmuka antara *game server online* dengan *compatible game client*. Secara garis besar kinerja website yang akan dibuat ialah menerima HTTP *request* beserta POST *method input* (bila ada) dari *game client* kemudian diteruskan ke *procedure* yang telah dibuat untuk diproses. Bilamana proses dalam *procedure* menghasilkan output, website bertugas pula meng-*encode output* tersebut hingga menjadi *response* dalam format JSON yang bakal diproses oleh *game client*.

3.1.4 Pengujian

Tujuan dilakukannya pengujian perangkat lunak pada penelitian ini ialah untuk mencari tahu sampai sejauh mana kemampuan atau kinerja perangkat lunak yang telah dikembangkan, apakah bisa beroperasi sebagaimana mestinya, sesuai dengan spesifikasi perancangan serta mendeteksi keberadaan *bug* di dalamnya. Proses pengujian dilakukan dengan menggunakan pengujian fungsional dan pengujian non-fungsional. Untuk pengujian fungsional meliputi whitebox testing dan blackbox testing. Sedangkan untuk pengujian non-fungsional akan digunakan validation testing. Gambar 3.4 menunjukkan diagram alir dalam proses implementasi.



Gambar 3.4 Diagram Alir Proses Pengujian

3.2 Perancangan

Perancangan permainan dalam penelitian ini terdiri dari 2 bagian yakni *Game Concept Design* dan *Technical Design*.

3.2.1 Game Concept Design

Main Game concepts atau konsep utama dari “Permainan *Strategy Kerajaan Majapahit*” dapat dilihat pada tabel 3.2.

Tabel 3.2 Main Game Concepts

No.	Elemen	Keterangan
1.	Judul <i>game</i>	Majapahit Wars
2.	<i>Game platform</i>	Android, Mac, Windows, Web
3.	<i>Target device</i>	Smartphone dan Desktop/ PC
4.	Target usia	Segala Usia
5.	<i>Genre game</i>	<i>Strategy</i>
6.	<i>USP(Unique Selling Point)</i>	1. <i>Multi platform</i>, atau dapat dimainkan berbagai macam <i>compatible game client</i> 2. <i>Multiplayer</i>, atau dapat dimainkan oleh beberapa orang 3. Merupakan permainan kompetitif yang mengharuskan para pemain berkompetisi secara sportif untuk menjadi yang terbaik 4. Memancing minat generasi muda untuk lebih peduli dan lebih mendalami materi sejarah Kerajaan Majapahit

3.2.1.1 Tujuan Permainan

Tujuan utama dari permainan ini ialah sebagai sarana hiburan kompetitif *multiplayer* serta *multi platform* yang sekaligus berfungsi sebagai sarana penarik minat generasi penerus bangsa agar lebih peduli terhadap sejarah warisan leluhur bangsa, khususnya sejarah Kerajaan Majapahit. Dalam permainan pemain akan berperan sebagai kepala desa suatu wilayah yang berada di bawah sistem pemerintahan Kerajaan Majapahit. Sebagai kepala desa, pemain dituntut untuk mengembangkan, memakmurkan serta menjaga keamanan dari wilayah yang dipimpinnya.

3.2.1.2 Kebutuhan Teknologi

Kebutuhan teknologi dan *tools* yang diperlukan guna mengembangkan permainan *strategy* Kerajaan Majapahit ditunjukkan pada tabel 3.3.

Tabel 3.3 Kebutuhan Teknologi

No	Kebutuhan	Kegunaan
1.	Aplikasi WAMP	Sebagai <i>web server</i> . PHP server dan HTTP server Apache sudah terintegrasi didalamnya.
2.	Aplikasi MySQL	Sebagai <i>database server</i> .
3.	Aplikasi MySQL Workbench	Sebagai alat pembantu pengkodean pada saat implementasi <i>gameplay</i> .
4.	Aplikasi Notepad++	Sebagai alat pembantu pengkodean pada saat implementasi <i>web</i> .
5.	Framework Code Igniter	Sebagai alat pembantu dalam implementasi <i>web</i> .
6.	Notebook atau laptop	Sebagai alat pengembangan perangkat lunak serta sebagai <i>host</i>

		untuk <i>game server</i> .

3.2.1.3 Kontrol permainan

Kontrol permainan atau *Game control* menunjukkan bagaimana mekanisme pengendalian permainan oleh pemain. Mekanisme pengendalian permainan dalam permainan *strategy* Kerajaan Majapahit bisa beraneka rupa, bergantung dari *compatible game client* dan *platform* yang digunakan oleh pemain. *Game server online* memberi kebebasan pada pengembang *game client* untuk mengatur mekanisme kontrol permainan pada aplikasi yang mereka buat namun tetap dalam koridor alur permainan yang disediakan *game server online*.

3.2.1.4 Jalan Cerita

Meskipun permainan ini mengambil latar belakang zaman serta wilayah dari Kerajaan Majapahit, namun jalan cerita permainan bersifat *non-linear* dan pada akhirnya para pemain yang nantinya akan membentuk jalan cerita dalam permainan ini. Para pemain dalam permainan diberi kebebasan dalam mengembangkan desa yang dipimpinnya serta mengadakan kontak dengan player atau desa lain. Arah pengembangan desa yang diterapkan oleh pemain dan bentuk kontak antar desa ialah pembentuk jalan cerita pada permainan *strategy* Kerajaan Majapahit.

3.2.1.5 Judul – Tampilan Awal

Seperti halnya dalam kontrol permainan, tampilan awal dari permainan *strategy* Kerajaan Majapahit bergantung dari *compatible game client* yang digunakan oleh pemain. Namun secara garis besar tampilan awal dari game ini memuat :

1. Pendaftaran :

Merupakan *form* pendaftaran yang harus dilengkapi oleh pemain untuk bisa bergabung dalam dunia permainan.

2. Masuk dunia permainan :

Menu yang digunakan pemain untuk memasuki dunia permainan.

3.2.1.6 Alur Permainan

Alur permainan pada permainan *strategy* Kerajaan Majapahit terdiri dari 6 bagian yakni :

1. Pendaftaran pemain

Sebelum dapat memasuki dunia permainan, pemain diharuskan mendaftarkan diri terlebih dahulu ke *game server online* dengan syarat *username* yang pemain pilih tidak boleh sama dengan *username* pemain lain. Manakala pendaftaran telah sukses, pemain akan diberi tanggung jawab untuk mengembangkan suatu desa di lokasi tertentu dengan dibekali sumber daya dalam jumlah tertentu dan sejumlah bangunan penghasil sumber daya.

2. Masuk dunia permainan

Guna menghindari penggunaan *username* oleh pemain lain, terutama pada *platform compatible game client* yang dipakai secara komunal maka diperlukan mekanisme dalam memasuki dunia permainan. Mekanisme yang dilakukan ialah pengecekan *username* dan *password* sebelum memasuki dunia permainan.

3. Pengumpulan sumber daya

Pada awal permainan, pengumpulan sumber daya dilakukan terjadwal per satuan waktu tertentu pada tiap bangunan penghasil sumber daya yang ada di desa pemain. Sumber daya bisa pula dikumpulkan melalui rampasan perang dengan cara memenangkan pertempuran manakala pasukan menyerang desa lain.

4. Pembangunan bangunan

Dari sumber daya yang dikumpulkan pemain bisa mengembangkan desanya dengan cara membangun berbagai macam bangunan di desa mereka. Secara garis besar ada 2 jenis bangunan dalam permainan ini yakni bangunan penghasil sumber daya dan bangunan non-penghasil sumber daya.

5. Pelatihan pasukan

Pasukan berfungsi sebagai penjaga keamanan desa dari serangan desa lain serta penyerbu dan pengumpul sumber daya rampasan perang ketika pemain memenangkan perang dengan desa lain.

6. Kontak dengan desa lain

Ada 2 macam kontak antar desa dalam permainan ini yakni menerima serangan dan menyerang desa lain. Kontak dengan desa lain bertujuan untuk mendapatkan sumber daya guna mengembangkan desa pemain.

3.2.1.7 Sumber Daya

Terdapat 3 jenis sumber daya pada permainan ini yakni makanan, kayu dan emas. Sumber daya berfungsi untuk mengembangkan desa pemain yang meliputi pembangunan bangunan dan pelatihan pasukan. Setiap pemain dibekali sejumlah sumber daya pada saat memulai permainan yang detailnya ditunjukkan pada tabel 3.4.

Tabel 3.4 Sumber Daya Awal

	Makanan	Kayu	Emas
Jumlah	300	300	100

Produksi sumber daya dilakukan terjadwal per satuan waktu tertentu pada tiap bangunan penghasil sumber daya. Detil produksi sumber daya melalui bangunan penghasil sumber daya ditunjukkan pada tabel 3.5.

Tabel 3. 5 Bangunan Penghasil Sumber Daya

	Kebun	Pemotongan Kayu	Tambang
Jenis Sumber Daya	Makanan	Kayu	Emas
Jumlah	15		
Waktu Produksi	Tiap 5 Menit		

Dalam game ini sumber daya bisa pula didapatkan dari hasil rampasan perang manakala pasukan yang dikirim untuk menyerang desa lawan mampu dan telah memenangkan peperangan. Jumlah dan jenis sumber daya yang diperoleh jumlahnya bergantung dari 2 faktor yakni jumlah sumber daya yang ada di desa terserang serta daya tampung sumber daya pasukan yang selamat dari peperangan.

3.2.1.8 Bangunan

Menurut fungsi dasarnya, dalam permainan ini terdapat 2 kategori bangunan yang bisa dibangun pada desa pemain. Kategori yang pertama ialah bangunan penghasil sumber daya yang nama, harga serta waktu pembangunannya bisa dilihat pada tabel 3.6.

Tabel 3. 6 Bangunan Penghasil Sumber Daya

	Makanan	Kayu	Emas	Waktu
Kebun	800	1200	1200	1 Jam

Pemotongan Kayu	1200	800	1200	
Tambang	1200	1200	800	

Kategori yang kedua ialah bangunan non-penghasil sumber daya. Dalam kategori ini hanya terdapat bangunan barak yang berfungsi sebagai tempat pelatihan pasukan penyerang dan pertahanan desa pemain. Harga serta waktu pembangunan barak bisa dilihat pada tabel 3.7.

Tabel 3.7 Bangunan Non Penghasil Sumber Daya

	Makanan	Kayu	Emas	Waktu
Barak	5000	8000	7000	1 Jam

Tiap desa diberikan 6 slot untuk melakukan pembangunan bangunan secara simultan.

3.2.1.9 Karakter permainan

Ada 2 kategori karakter dalam permainan ini. Kategori karakter yang dimaksud ialah sebagai berikut :

1. Kepala Desa

Kepala desa ialah karakter utama permainan yang diperankan serta dikontrol secara langsung oleh pemain. Kepala desa dalam permainan ini bertugas mengembangkan serta mengambil arah kebijakan terhadap wilayah yang dipimpinnya, baik dalam kebijakan militer maupun non-militer. Arah kebijakan kepala desa akan sangat berpengaruh terhadap keberhasilan wilayah yang dipimpinnya.

2. Pasukan

Pasukan ialah karakter dalam permainan yang bertugas menyerang dan mengumpulkan sumber daya dari desa lain sekaligus mempertahankan wilayah desa tatkala ada serangan dari desa lain. Terdapat 2 jenis psukan dalam permainan ini yakni Kris Swordman dan Bhayangkara. Biaya serta waktu pelatihan 2 pasukan tersebut bisa dilihat pada tabel 3.8.

Tabel 3.8 Biaya & Waktu Pelatihan Pasukan

	Makanan	Kayu	Emas	Waktu
Kris Swordman	300	0	500	2 Jam
Bhayangkara	0	200	300	

Bilamana pemain hendak membangun pasukan, pemain diharuskan mempunyai atau membangun minimal 1 barak terlebih dahulu. Tiap barak mempunyai 10 slot untuk pelatihan pasukan secara simultan. Pasukan dalam permainan ini memiliki 4 atribut yang berpengaruh manakala terjadi kontak dengan desa lain yakni AP(*attack point*), DP(*defense point*), CC(*carrying capacity*) dan TS(*troop speed*). Nilai atribut tiap pasukan bisa dilihat pada tabel 3.9.

Tabel 3.9 Atribut Pasukan

	AP	DP	CC	TS
Kris Swordman	300	100	20	15
Bhayangkara	100	300	40	20

3.2.1.10 Kamera Permainan

Jenis kamera permainan dalam permainan *strategy* Kerajaan Majapahit bisa bermacam-macam, bergantung dari *compatible game client* yang digunakan oleh pemain. *Game server online* memberi kebebasan pengembang *game client* memilih serta mengatur kamera permainan sesuai kreativitas mereka.

3.2.1.11 Sistem HUD

Guna mendukung alur permainan, informasi minimal yang harus ditampilkan di *game client* ialah jumlah sumber daya yang dimiliki pemain, daftar pasukan pertahanan desa, jadwal serangan ke/dari desa lain, jadwal kepulangan pasukan serta daftar *ranking* pemain. Penempatan informasi bebas, sesuai kreativitas pengembang *game client*.

3.2.1.12 Ranking

Ranking dalam permainan diambil dari pengurutan *point* para pemain yang nilainya merupakan penjumlahan dari total keseluruhan sumber daya dengan total keseluruhan pasukan. *Point* yang dimiliki pemain bersifat dinamis dan perubahannya akan selalu *real-time* bergantung dari perubahan jumlah sumber daya dan perubahan jumlah pasukan pemain.

3.2.1.13 Tampilan dunia

Tampilan dunia dalam permainan *strategy* Kerajaan Majapahit sangat relatif, bergantung dari *compatible game client* yang dipakai. Tampilan dunia *game client* harus dapat mengakomodasi alur permainan *strategy* Kerajaan Majapahit serta diharapkan mengandung tema Kerajaan Majapahit dalam desainnya.

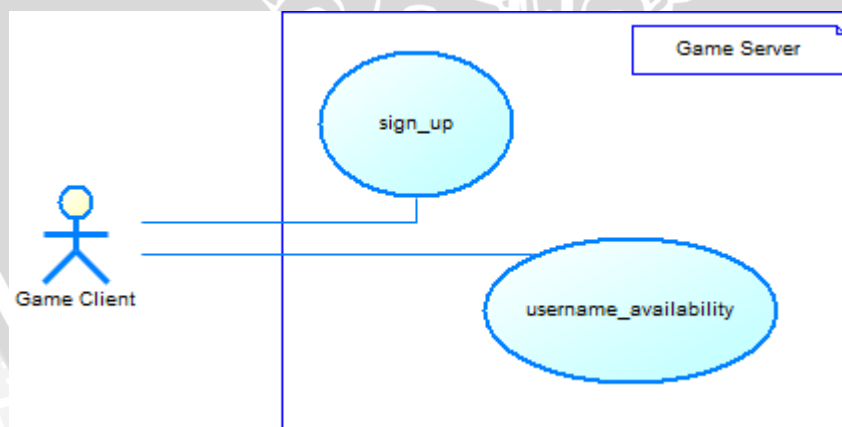
3.2.2 Technical Design

Pada tahap *technical design* atau perancangan teknis, perancangan sistem dilakukan dengan pemodelan menggunakan UML (*Unified Modelling Language*). Pemodelan yang dirancang ialah pemodelan *game server online* dari permainan *strategy* kerajaan Majapahit.

3.2.2.1 Use Case Diagram

Diagram *use case* dalam perancangan *game server online* permainan *strategy* Kerajaan Majapahit digunakan untuk memodelkan fungsionalitas permainan yang disusun dengan berpedoman atau mengacu dari dokumen *game concept design*. Aktor yang berperan dalam *use case diagram* dari *game server online* ialah *game client*. Guna mempermudah dalam perancangan dan implementasi pada *game client*, diagram *use case* akan dibuat berdasarkan alur permainan yang telah dibuat. Diagram *use case* dari sistem *game server online* permainan *strategy* Kerajaan Majapahit ialah sebagai berikut :

1. Pendaftaran Pemain



Gambar 3.5 Use Case Pendaftaran Pemain

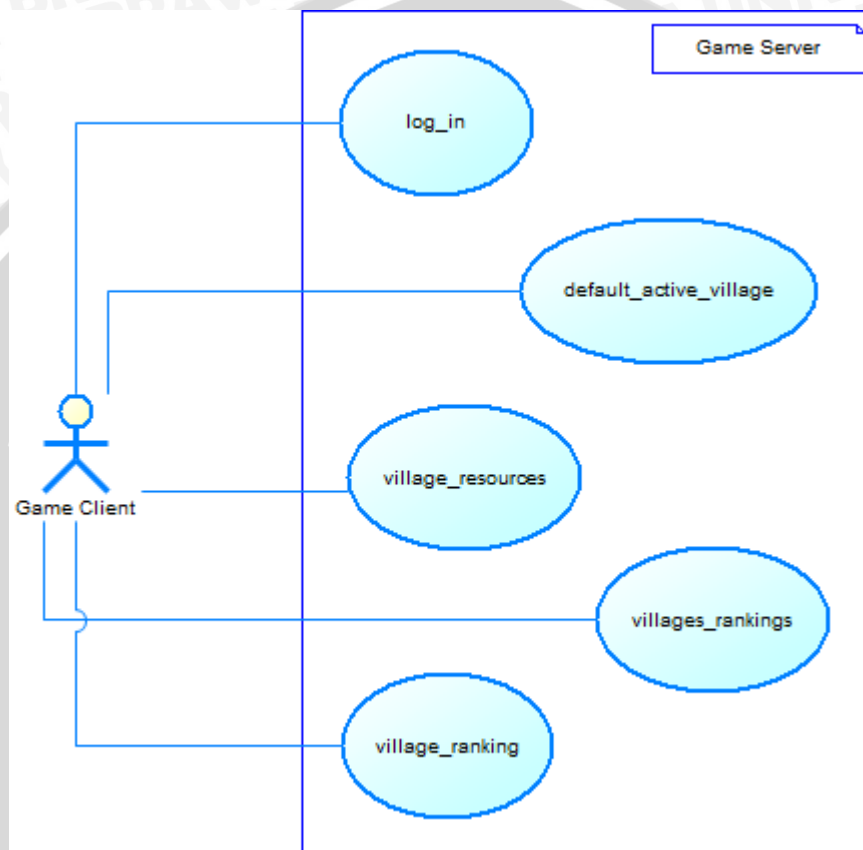
a. sign_up

Berfungsi untuk mendaftarkan *username* dan *password* yang pemain pilih manakala pemain melakukan pendaftaran pada *game server*.

b. username_availability

Digunakan *game client* untuk memvalidasi *username* yang dipilih pemain sebelum dilakukan pendaftaran dengan fungsi *sign_up*.

2. Masuk dunia permainan



Gambar 3.6 Use Case Masuk Dunia Permainan

a. *log_in*

Berguna dalam memverifikasi *username* dan *password* pemain sebelum memasuki dunia permainan.

b. *default_active_village*

Berfungsi untuk mendapatkan ID dari desa pemain.

c. *village_ranking*

Digunakan untuk menampilkan peringkat serta nilai point yang dimiliki oleh desa pemain.

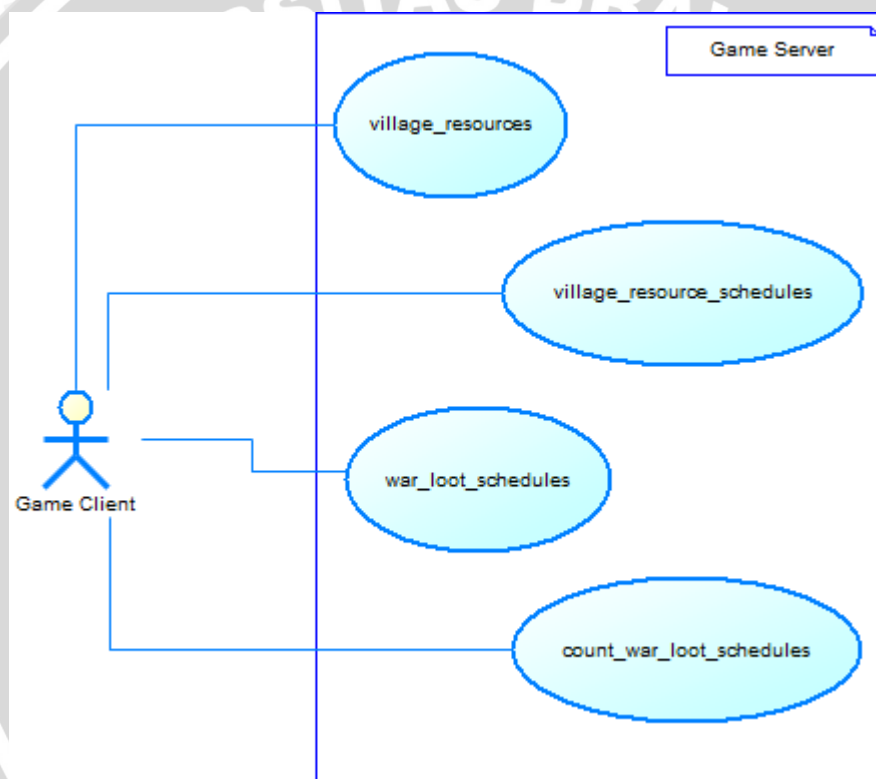
d. villages_rankings

Berfungsi untuk menampilkan daftar ranking dan daftar point dari keseluruhan pemain yang terdaftar pada game server.

e. village_resources

Berfungsi untuk mengetahui detil jumlah sumber daya yang dimiliki oleh desa pemain.

3. Pengumpulan sumber daya



Gambar 3.7 Use Case Pengumpulan Sumber Daya

a. village_resources

Berfungsi untuk mengetahui detil jumlah sumber daya yang dimiliki oleh desa pemain.

b. village_resource_schedules

Berfungsi untuk menampilkan daftar jadwal produksi sumber daya dari bangunan penghasil sumber daya yang dimiliki desa pemain.

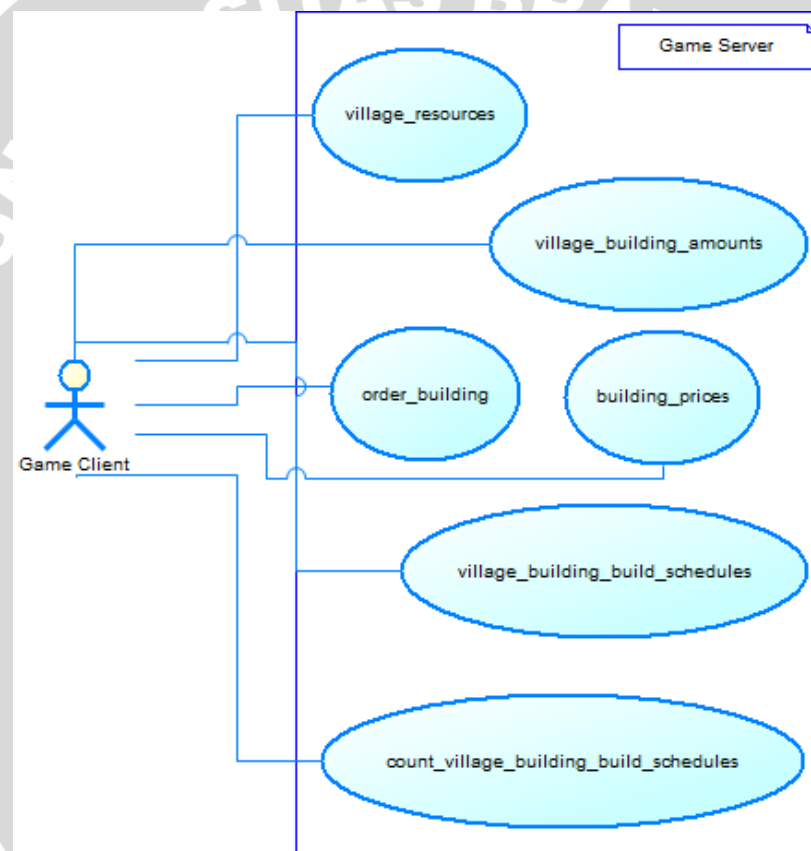
c. war_loot_schedules

Digunakan untuk menampilkan daftar jadwal tiba sumber daya yang didapat dari rampasan perang.

d. count_war_loot_schedules

Untuk menampilkan jumlah daftar jadwal yang ditampilkan fungsi war_loot_schedules.

4. Pembangunan bangunan



Gambar 3.8 Use Case Pembangunan Bangunan

a. village_resources

Berfungsi untuk mengetahui detail jumlah sumber daya yang dimiliki oleh desa pemain.

b. village_building_amounts

Berfungsi untuk menampilkan jumlah bangunan tertentu yang dimiliki oleh desa pemain.

c. `order_building`

Fungsi yang digunakan pemain untuk membangun suatu bangunan di desa pemain.

d. `building_prices`

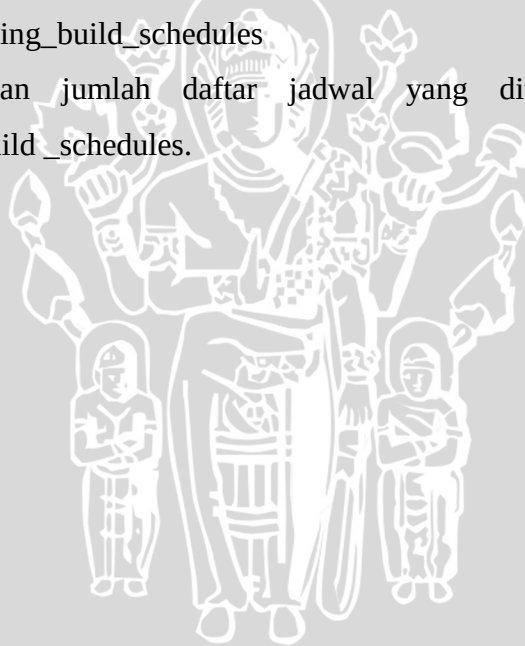
Berfungsi untuk menampilkan daftar bangunan yang bisa dibangun lengkap beserta sumber daya yang diperlukan untuk memulai pembangunan.

e. `village_building_build_schedules`

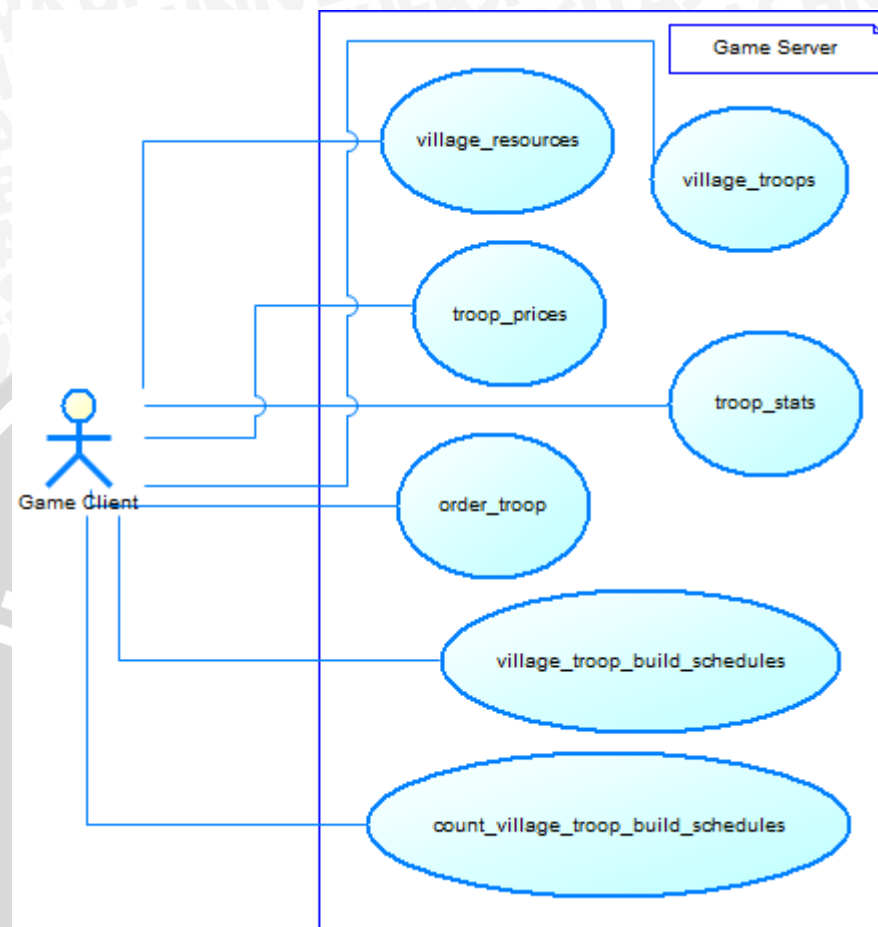
Digunakan untuk menampilkan daftar jadwal pembangunan bangunan yang sedang berlangsung dalam desa pemain.

f. `count_village_building_build_schedules`

Untuk menampilkan jumlah daftar jadwal yang ditampilkan fungsi `village_building_build_schedules`.



5. Pelatihan pasukan



Gambar 3.9 Use Case Pelatihan Pasukan

a. village_resources

Berfungsi untuk mengetahui detail jumlah sumber daya yang dimiliki oleh desa pemain.

b. village_troops

Berfungsi untuk menampilkan jenis serta jumlah pasukan yang sedang berada di desa pemain.

c. order_troop

Fungsi yang digunakan pemain untuk melatih pasukan.

d. troop_stats

Digunakan untuk menampilkan daftar pasukan yang bisa dilatih lengkap beserta informasi atribut yang dimilikinya.

e. troop_prices

Berfungsi untuk menampilkan daftar pasukan yang bisa dilatih lengkap beserta sumber daya yang diperlukan untuk memulai pelatihan.

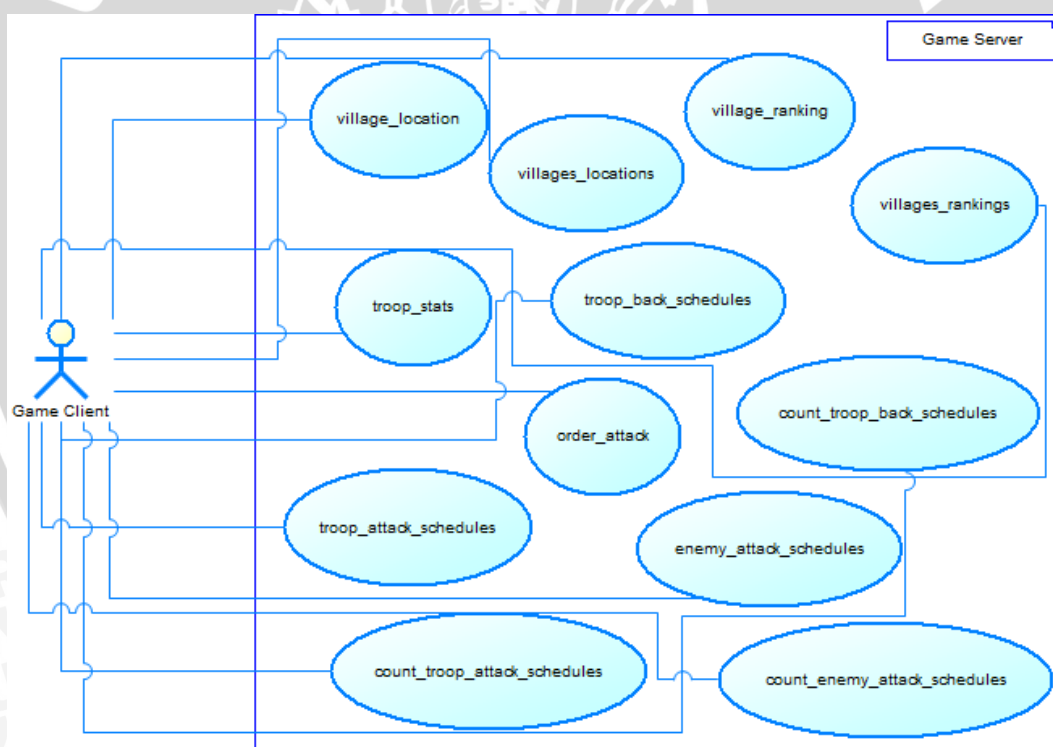
f. village_troop_build_schedules

Digunakan untuk menampilkan daftar jadwal pelatihan pasukan yang sedang berlangsung dalam desa pemain.

g. count_village_troop_build_schedules

Untuk menampilkan jumlah daftar jadwal yang ditampilkan fungsi village_troop_build_schedules.

6. Kontak dengan desa lain



Gambar 3.10 Use Case Kontak dengan Desa Lain

a. village_troops

Berfungsi untuk menampilkan jenis serta jumlah pasukan yang sedang berada di desa pemain.

b. `order_attack`

Fungsi yang digunakan pemain untuk mengirim pasukan guna menyerang desa pemain lain.

c. `troop_stats`

Digunakan untuk menampilkan daftar pasukan yang bisa dilatih lengkap beserta informasi atribut yang dimilikinya.

d. `village_ranking`

Digunakan untuk menampilkan peringkat serta nilai point yang dimiliki oleh desa pemain.

e. `villages_rankings`

Berfungsi untuk menampilkan daftar ranking dan daftar point dari keseluruhan pemain yang terdaftar pada game server.

f. `village_location`

Digunakan untuk menampilkan koordinat lokasi desa pemain.

g. `villages_locations`

Berfungsi untuk menampilkan koordinat lokasi serta informasi waktu tempuh pasukan pemain terhadap keseluruhan desa pemain yang terdaftar pada game server.

h. `troop_attack_schedules`

Digunakan untuk menampilkan daftar jadwal serangan pasukan desa pemain terhadap desa pemain lain.

i. `count_troop_attack_schedules`

Untuk menampilkan jumlah daftar jadwal yang ditampilkan fungsi `troop_attack_schedules`.

j. `troop_back_schedules`

Digunakan untuk menampilkan daftar jadwal kembalinya pasukan desa pemain yang telah memenangkan peperangan dengan desa pemain lain.

k. `count_troop_back_schedules`

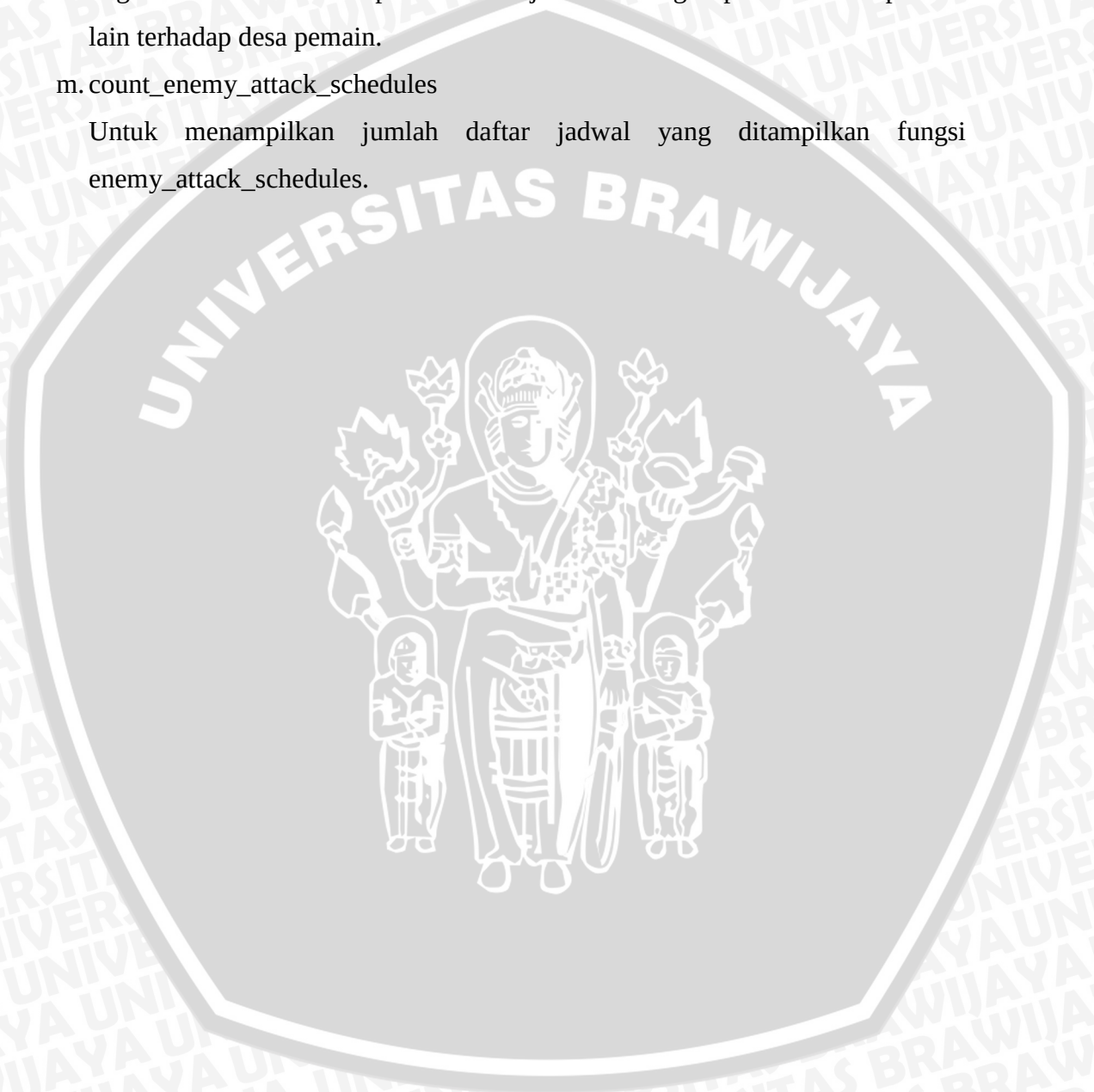
Untuk menampilkan jumlah daftar jadwal yang ditampilkan fungsi `troop_back_schedules`.

l. `enemy_attack_schedules`

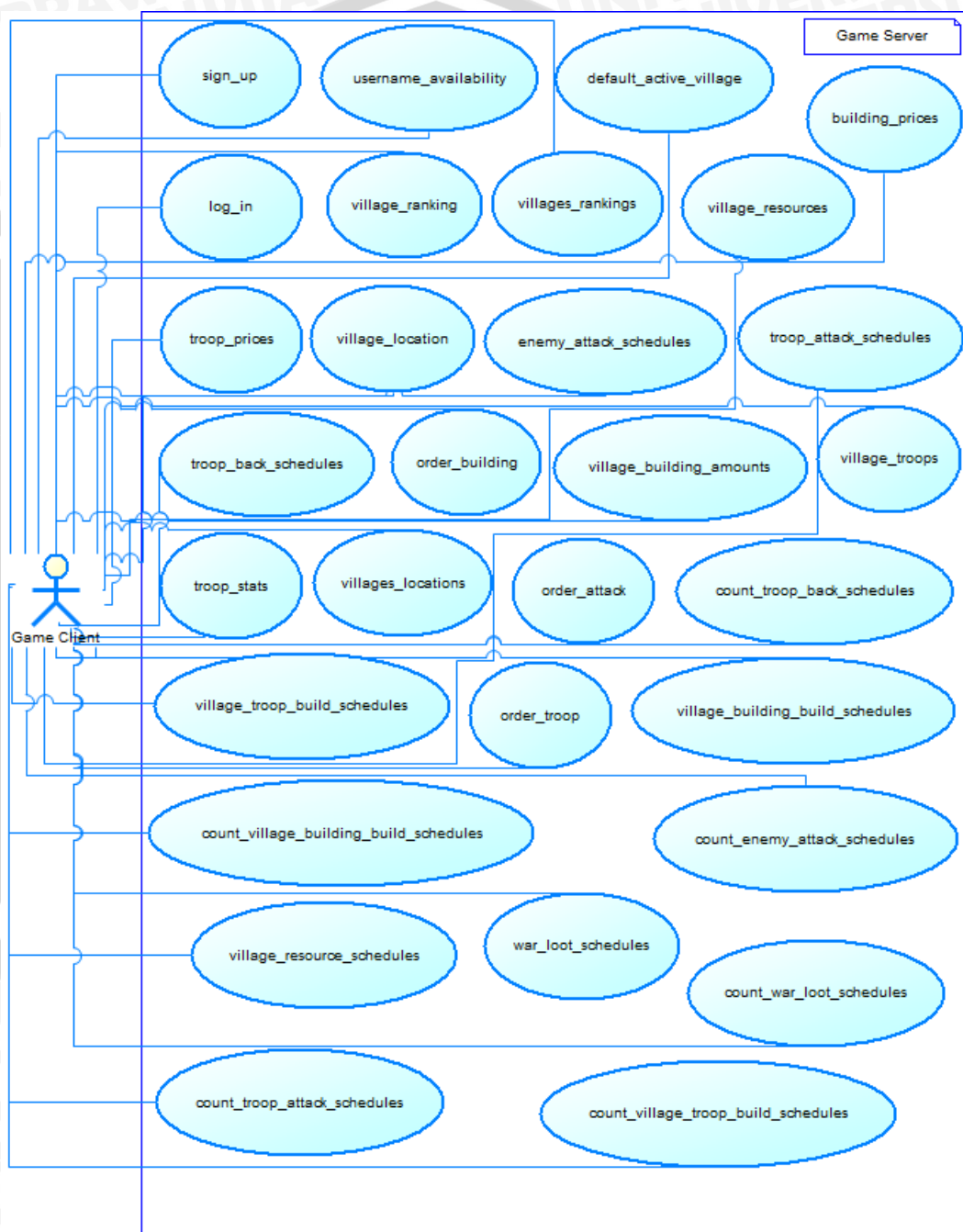
Digunakan untuk menampilkan daftar jadwal serangan pasukan desa pemain lain terhadap desa pemain.

m. `count_enemy_attack_schedules`

Untuk menampilkan jumlah daftar jadwal yang ditampilkan fungsi `enemy_attack_schedules`.



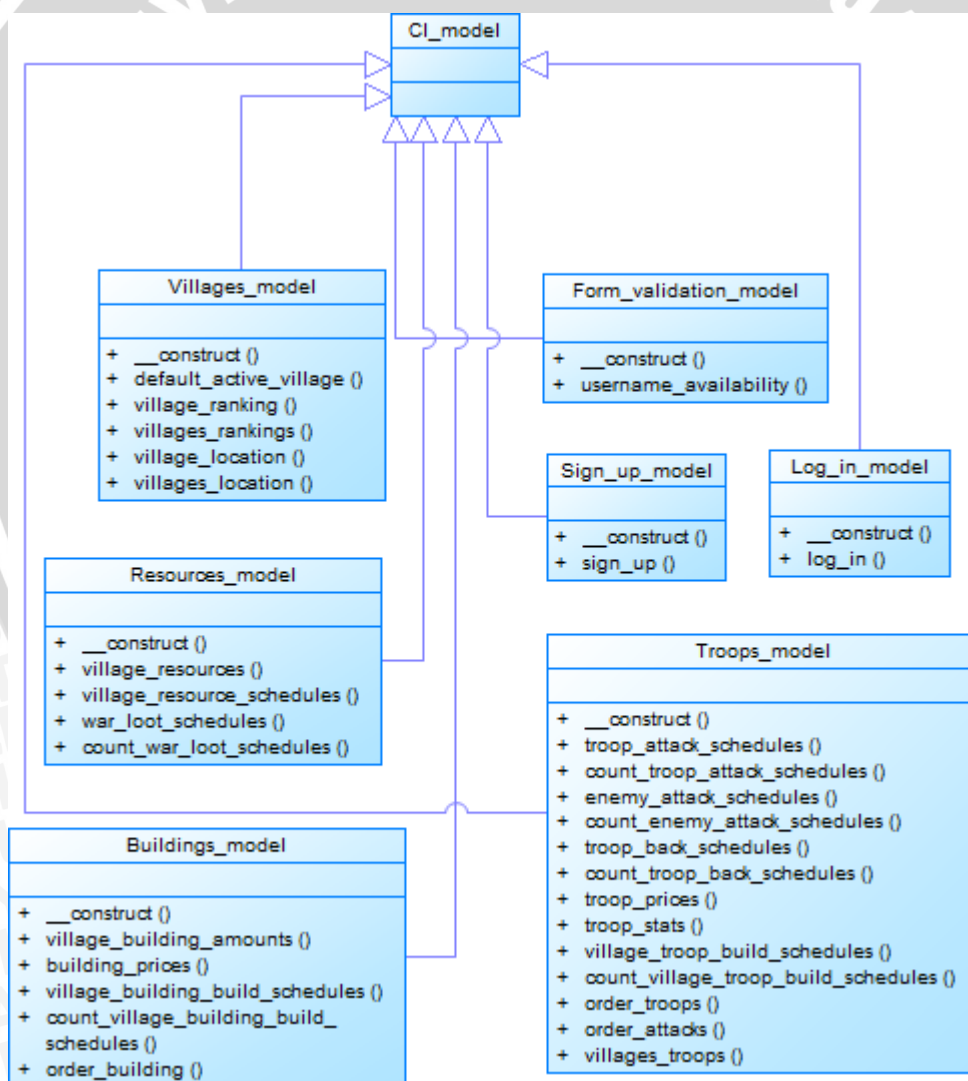
Dari diagram *use case* yang telah dibuat berdasarkan alur permainan, bisa disusun diagram *use case* keseluruhan dari sistem *game server online* permainan *strategy* Kerajaan Majapahit yang ditunjukkan pada gambar 3.9.



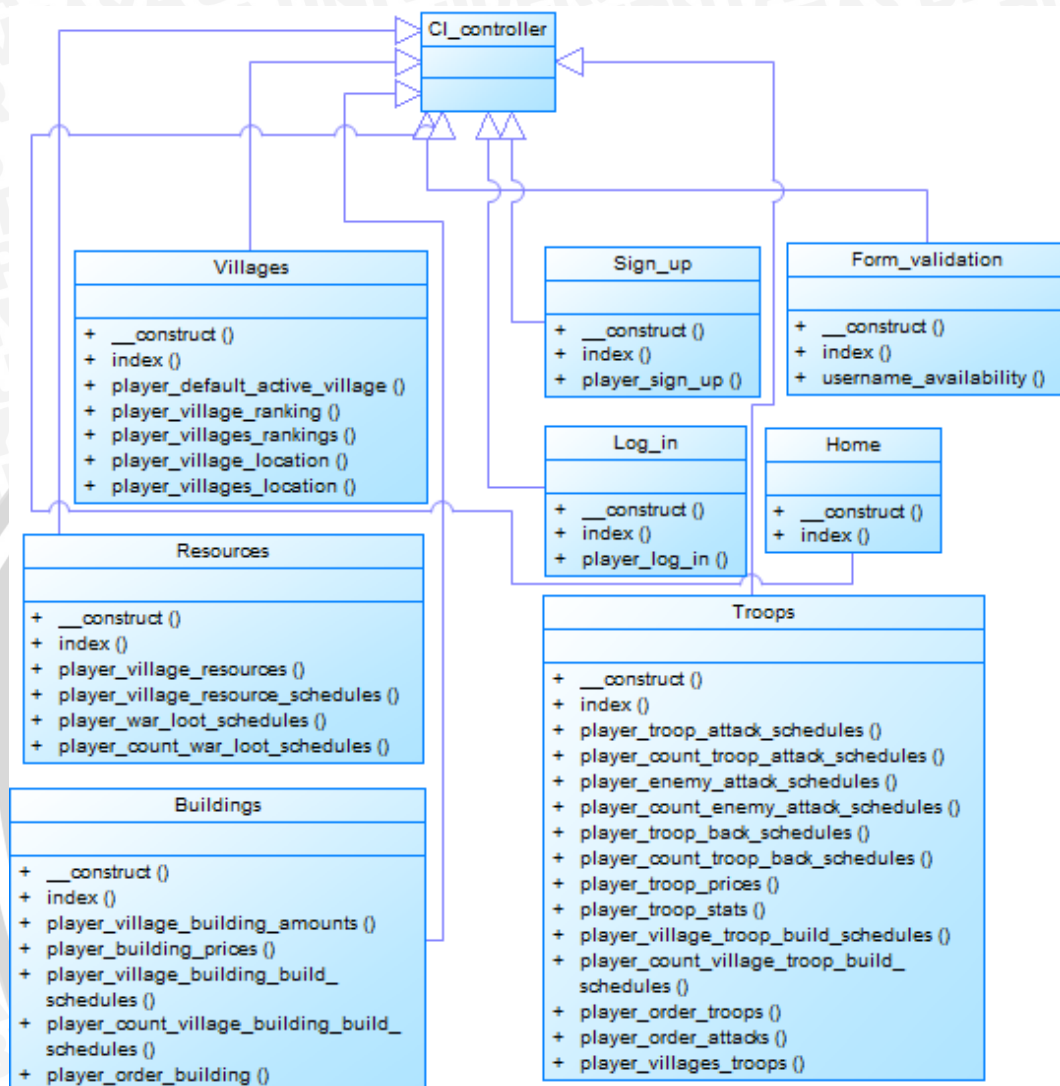
Gambar 3.11 Use Case Permainan Strategy Kerajaan Majapahit

3.2.2.2 Class Diagram

Perancangan diagram kelas bertujuan untuk memodelkan kelas-kelas dan *interface-interface* yang dibutuhkan pada saat proses implementasi. Perancangan diagram kelas berpedoman atau mengacu pada *design pattern* MVC *framework* Code Igniter serta pengelompokan atau klasifikasi fungsi-fungsi dalam diagram *use case* berdasarkan objek-objek permainan yang bertujuan untuk mempermudah saat melakukan pengembangan *game client*. Terdapat 2 diagram kelas dalam perancangan yakni diagram kelas model yang ditunjukkan pada gambar 3.10 dan diagram kelas controller yang ditunjukkan pada gambar 3.11.



Gambar 3.12 Class Diagram Model



Gambar 3.13 Class Diagram Controller

Diagram kelas model memodelkan kelas-kelas yang berhubungan langsung dengan fungsi-fungsi permainan dalam database dan diagram kelas controller memodelkan kelas-kelas yang nantinya akan berhubungan langsung dengan *game client*. *Game client* tidak dapat mengakses langsung kelas-kelas yang ada pada diagram kelas model. Pengkasesan kelas pada diagram kelas model harus melalui kelas-kelas yang disediakan pada diagram kelas controller.

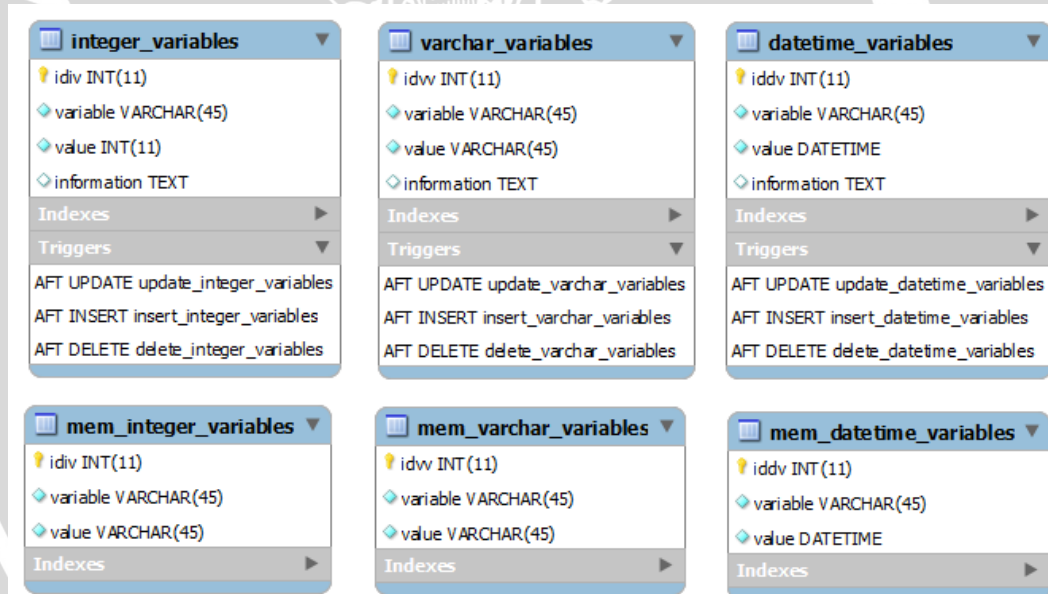
3.2.2.3 EER Diagram

Perancangan *Enhanced Entity-Relationship diagram* atau diagram EER bertujuan untuk memodelkan bentuk database dari perangkat lunak permainan yang mencakup *table* lengkap dengan *trigger* beserta relasi diantaranya dan *view*. Diagram EER dari sistem *game server online* permainan *strategy* Kerajaan Majapahit ialah sebagai berikut :

1. Table

Guna mempermudah dalam pengembangan, diagram EER untuk table akan dibuat perobjek yakni :

a. Variables

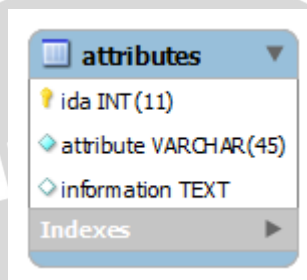


Gambar 3.14 EER Diagram Variables

Table *mem_integer_variables*, *table* *mem_varchar_variables* dan *table* *mem_datetime_variables* ialah *table-table* dengan *memory engine* yang merupakan *clone* atau duplikat dari *table-table variables*. Perbedaan *engine* mengakibatkan tidak dimungkinkan adanya relasi antara *table-table variables*

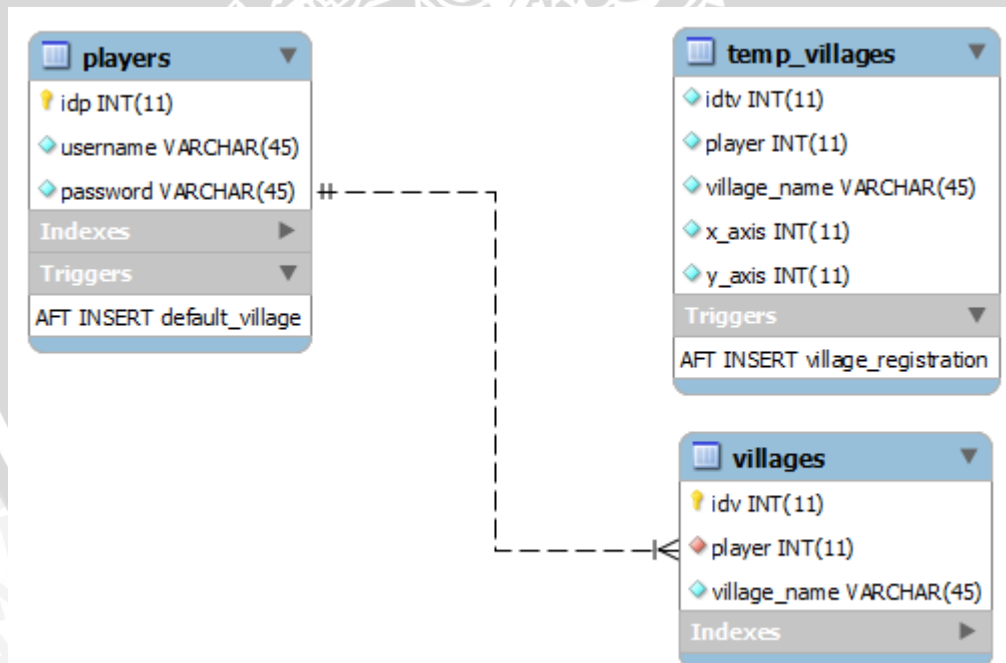
dengan *table-table clone*-nya. Pengaksesan *variables* oleh *procedure*, *function* dan *view* dalam perangkat lunak *game server online* akan menggunakan table dengan memory engine.

b. Attributes



Gambar 3.15 EER Diagram Attributes

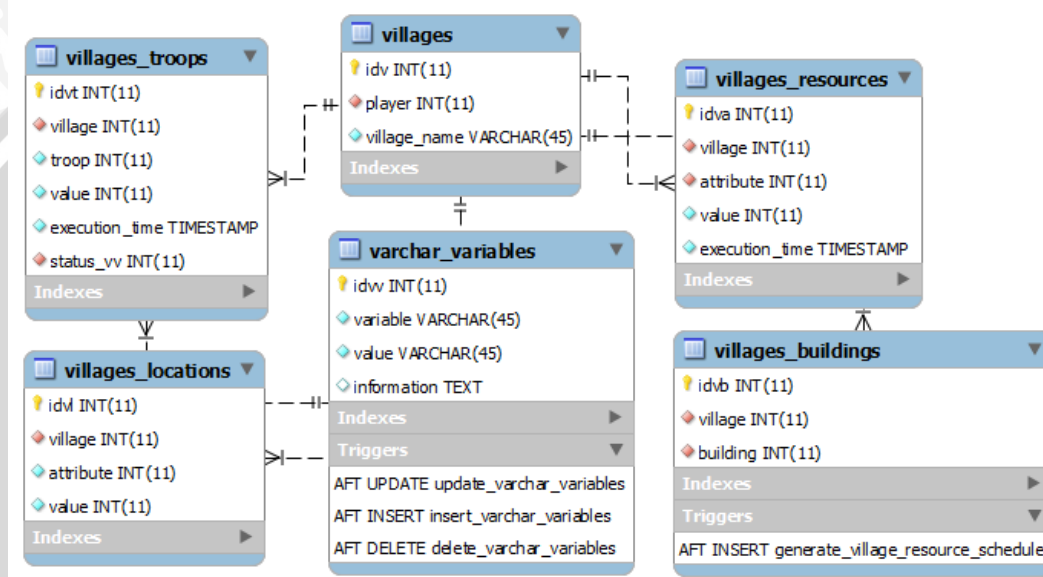
c. Players



Gambar 3.16 EER Diagram Players

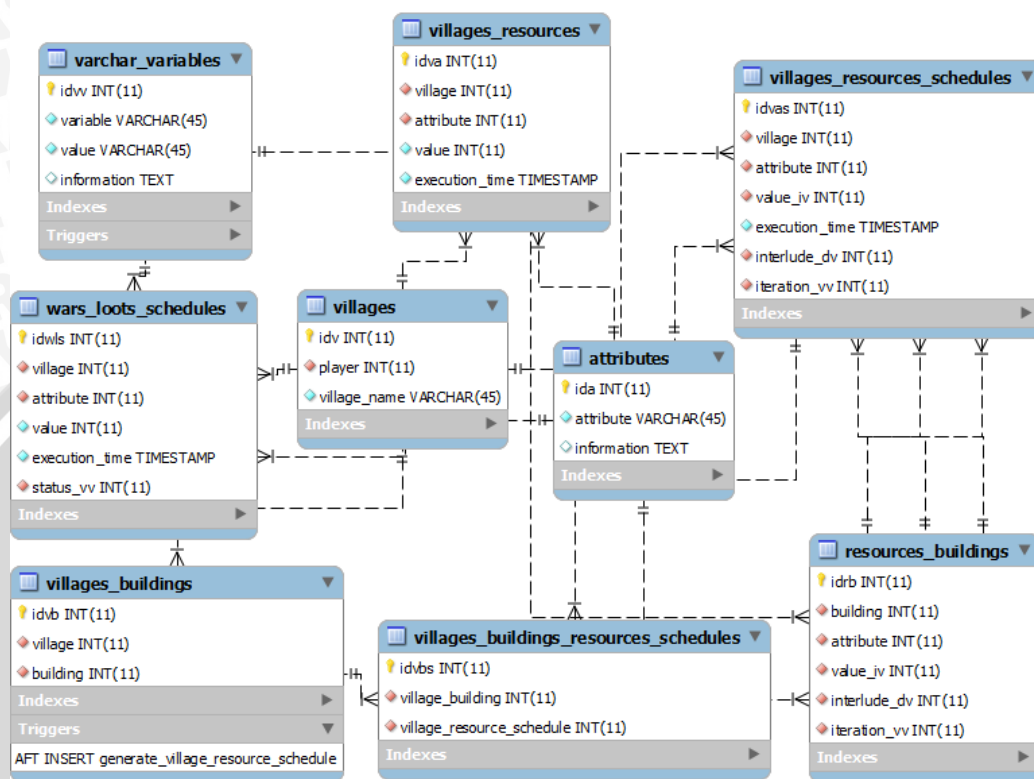
Table temp_villages merupakan table dengan blackhole engine. Table dengan Blackhole engine tidak dapat menyimpan data namun dapat digunakan untuk mentransfer parameter yang tidak dimungkinkan manakala hanya menggunakan trigger.

d. Villages



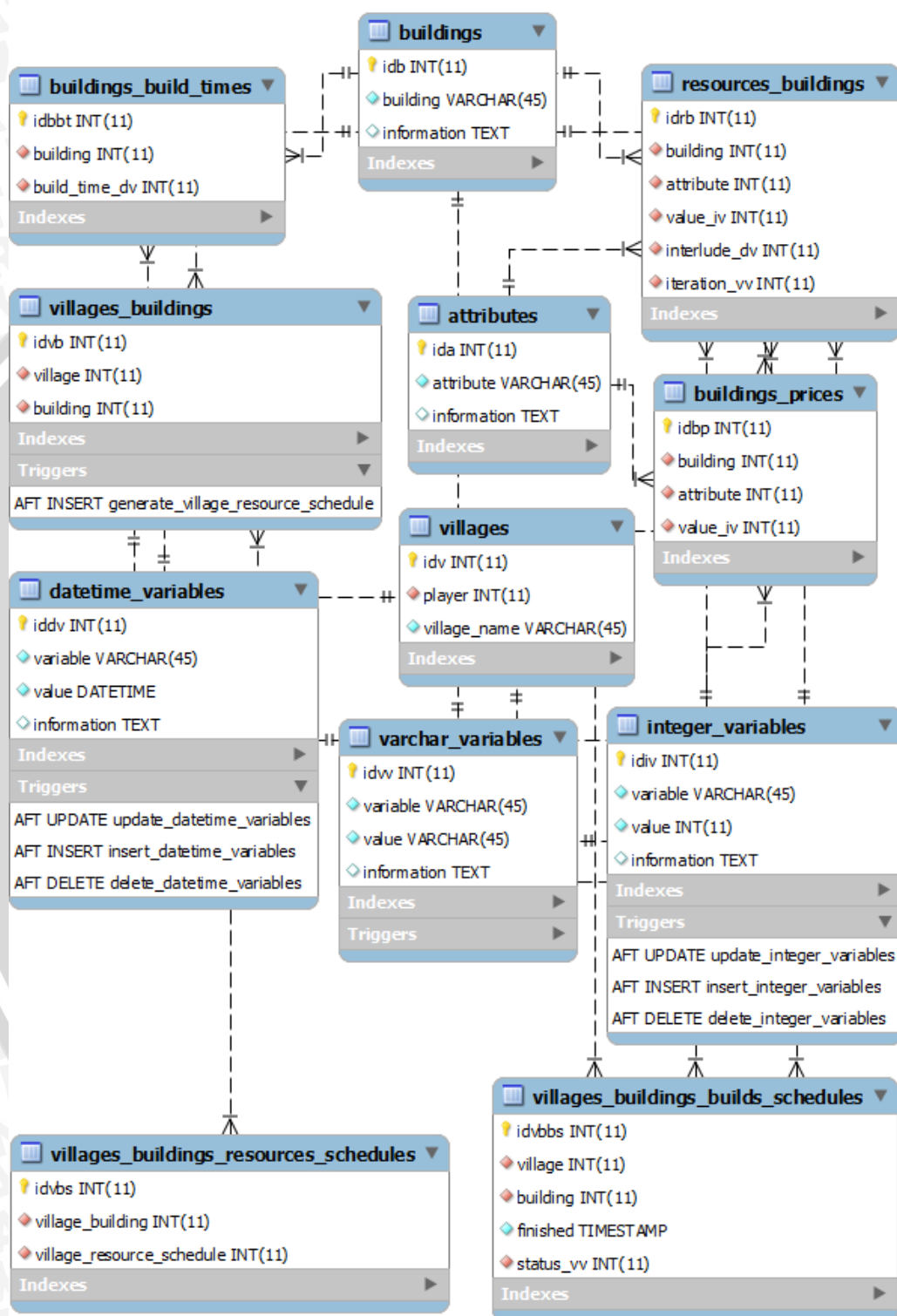
Gambar 3.17 EER Diagram Villages

e. Resources



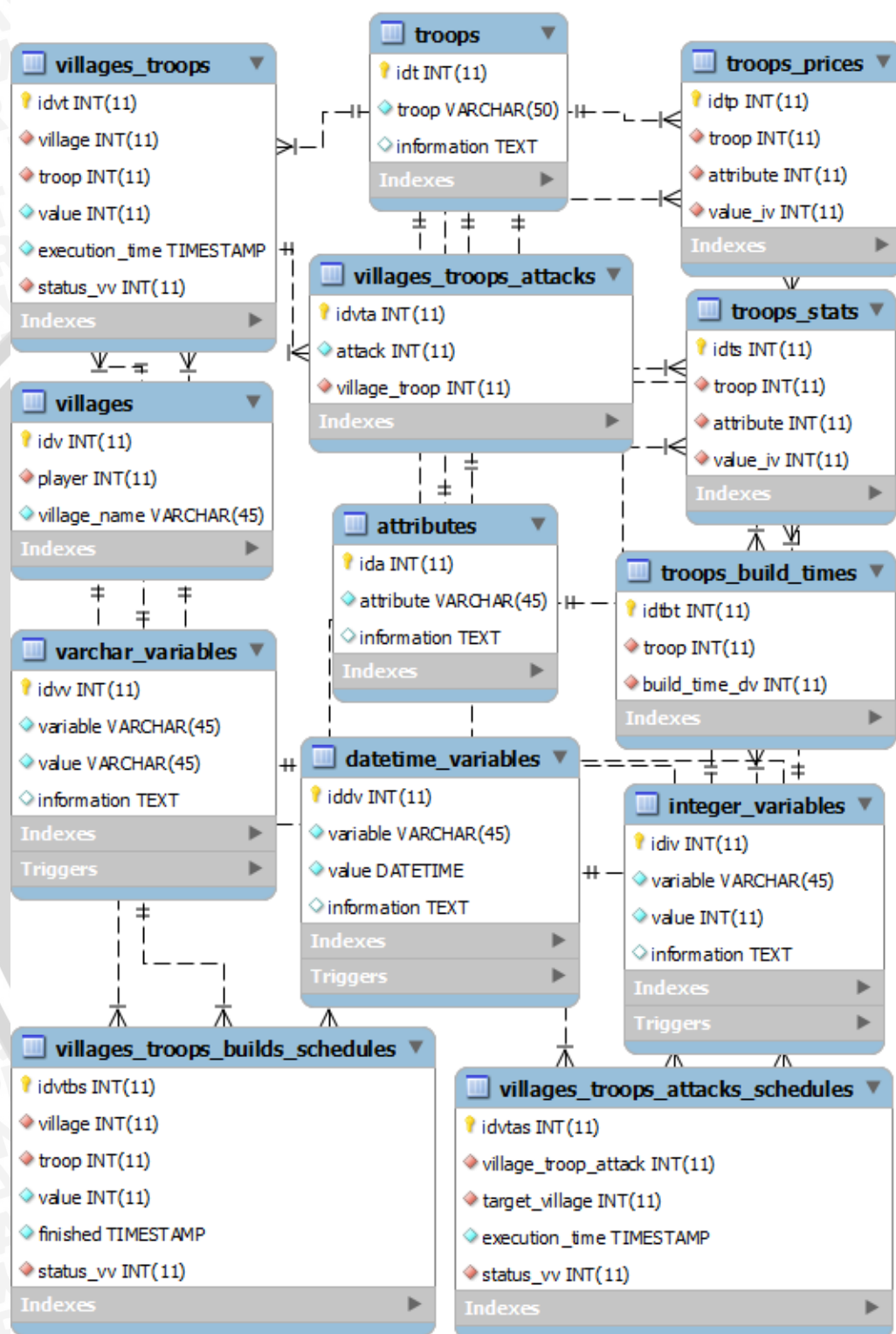
Gambar 3.18 EER Diagram Resources

f. Buildings



Gambar 3.19 EER Diagram Buildings

g. Troops



Gambar 3.20 EER Diagram Troops

2. Views

Diagram EER untuk *view* yang dirancang dalam pengembangan *game server online* permainan *strategy* Kerajaan Majapahit ditunjukkan pada gambar 3.17.



Gambar 3.21 EER Diagram Views

BAB IV

IMPLEMENTASI

Dalam bab ini akan dibahas tentang proses implementasi secara menyeluruh dari pengembangan *game server online* permainan *strategy* Kerajaan Majapahit dengan mengacu atau berpedoman pada dokumen-dokumen perancangan permainan yang telah dibuat.

4.1 Spesifikasi Sistem

Perangkat lunak *game server online* permainan *strategy* Kerajaan Majapahit dikembangkan dalam *environment* atau lingkungan implementasi yang terdiri dari perangkat keras dan perangkat lunak.

4.1.1 Spesifikasi Lingkungan Perangkat Keras

Perangkat keras yang dipakai dalam proses pengembangan perangkat lunak ialah laptop atau *notebook*. Spesifikasi dari laptop yang dipakai dijelaskan pada tabel 4.1.

Tabel 4.1 Spesifikasi Lingkungan Perangkat Keras Laptop

DELL INSPIRON 14	
<i>Processor</i>	Intel(R) Core(TM) i3-2365M CPU @ 1.40GHz (4 CPUs), ~1.4GHz
<i>Memory (RAM)</i>	2 x Kingston 2 GB DDR3 PC3-12800 (800MHz)
<i>Harddisk Drive</i>	Western Digital Scorpio Blue 500GB Internal Western Digital My Book 4TB External
<i>Motherboard</i>	DELL Inc. 046D87 A02 Sany Bridge HM76
<i>Graphic Card</i>	Intel(R) HD Graphics 3000 64MB Dedicated NVIDIA GeForce GT 625M 1024MB Dedicated

4.1.2 Spesifikasi Lingkungan Perangkat Lunak

Spesifikasi perangkat lunak yang dipakai dalam proses pengembangan *game server online* terdiri dari perangkat lunak pengkodean yang dijelaskan pada tabel 4.2 dan perangkat lunak *server* yang dijelaskan pada tabel 4.3.

Tabel 4.2 Spesifikasi Lingkungan Perangkat Lunak Pengkodean

PERANGKAT LUNAK PENGKODEAN	
<i>Operating System</i>	Microsoft Windows 7 Ultimate 64-bit
<i>Server-Side Programming Language</i>	PHP
<i>Database Programming Language</i>	SQL
<i>Software Development Kit</i>	Code Igniter 2.1.4
<i>Database Design Tool</i>	MySQL Workbench 5.2.47 CE
<i>Code Editor</i>	Notepad++ 6.6.3

Tabel 4.3 Spesifikasi Lingkungan Perangkat Lunak Server

PERANGKAT LUNAK SERVER	
<i>Web Environment</i>	WAMP Server 2.4
<i>HTTP Server</i>	Apache 2.4.9
<i>PHP Server</i>	PHP 5.5.12
<i>Database Server</i>	MySQL 5.6.17
<i>Database Administration</i>	PHPMysqlAdmin 4.1.14

4.2 Batasan-Batasan Dalam Implementasi

Beberapa batasan yang diperlukan saat dilakukannya implementasi perangkat lunak *game server online* permainan *strategy* Kerajaan Majapahit ialah sebagai berikut:

1. Implementasi algoritma *gameplay* dilakukan sepenuhnya dengan bahasa *query* pada MySQL.
2. Pengaksesan *table* dan *view* database oleh aplikasi *web* yang dibangun saat implementasi *web*, baik operasi SELECT, UPDATE maupun INSERT dilakukan melalui *procedure-procedure* yang dibuat sebelumnya pada implementasi algoritma *gameplay*.
3. Metode pemasukan data melalui HTTP *request* pada aplikasi *web* yang dibangun saat implementasi *web* menggunakan metode POST
4. Format keluaran data atau *response* dari aplikasi *web* yang dibangun saat implementasi *web* ialah format JSON
5. Pembangunan aplikasi *web* yang dibangun saat implementasi *web* dengan dibantu MVC *framework Code Igniter*.

4.3 Implementasi Algoritma *Gameplay*

Implementasi algoritma *gameplay* yang dilakukan sepenuhnya dengan pengkodean *query* dalam MySQL terdiri dari berbagai ragam algoritma *function*, *procedure* serta *event*. Dalam subbab ini akan dijelaskan algoritma dari keseluruhan *function*, *procedure* serta *event* yang ada dalam *game server online* permainan *strategy* Kerajaan Majapahit. Adapun ragam algoritma tersebut ialah:

4.3.1 Function

Ragam algoritma *function* dalam *game server online* permainan *strategy* Kerajaan Majapahit ialah sebagai berikut.

4.3.1.1 Implementasi Function *get_attribute_id*

Function yang digunakan untuk mendapatkan ID dari atribut yang diperlukan ketika pemrosesan. Pemrosesan dalam perangkat lunak yang dikembangkan akan lebih banyak menggunakan ID atribut daripada nama atribut.

Tabel 4.4 Implementasi Function *get_attribute_id*

Pseudocode Function <i>get_attribute_id</i>

Deklarasi awal

```
var_attribute_name:VARCHAR(45) -> Masukan dari user/sistem
var_ida:INT
```

Deskripsi

1	MULAI
2	
3	SELECT ida INTO var_ida
4	FROM attributes
5	WHERE attribute = var_attribute_name;
6	
7	RETURN var_ida;
8	
9	SELESAI

Penjelasan dari *function* `get_attribute_id` pada tabel 4.4 yaitu:

1. Keluaran dari *function* dalam tipe data INT sesuai dengan tipe data ID atribut pada tabel attribute.
2. Masukan *function* ialah nama atribut seperti food, wood atau gold.

4.3.1.2 Implementasi Function `get_building_id`

Function yang digunakan untuk mendapatkan ID dari bangunan yang diperlukan ketika pemrosesan. Pemrosesan dalam perangkat lunak yang dikembangkan akan lebih banyak menggunakan ID bangunan daripada nama bangunan.

Tabel 4.5 Implementasi Function `get_building_id`

Pseudocode Function <code>get_building_id</code>
Deklarasi awal <pre>var_building:VARCHAR(45) -> Masukan dari user/sistem var_idb:INT</pre>

Deskripsi	
1	MULAI
2	
3	SELECT idb INTO var_idb
4	FROM buildings
5	WHERE building = var_building;
6	
7	RETURN var_idb;
8	
9	SELESAI

Penjelasan dari *function* get_building_id pada tabel 4.5 yaitu:

1. Keluaran dari *function* dalam tipe data INT sesuai dengan tipe data ID bangunan pada tabel buildings.
2. Masukan *function* ialah nama bangunan seperti farm, lumber_camp atau mining_camp.

4.3.1.3 Implementasi Function get_datetime_variables

Function yang digunakan untuk mendapatkan nilai dari variabel dengan tipe data datetime pada tabel mem_datetime_variables.

Tabel 4.6 Pseudocode Function get_datetime_variables

Pseudocode Function get_datetime_variables	
Deklarasi awal	
var_variable:VARCHAR(45) -> Masukan dari user/sistem	
var_value:DATETIME	
Deskripsi	
1	MULAI
2	
3	SELECT value INTO var_value
4	FROM mem_datetime_variables
5	WHERE variable = var_variable;
6	

7	RETURN var_value;
8	
9	SELESAI

Penjelasan dari *function* `get_datetime_variables` pada tabel 4.6 yaitu:

1. Keluaran dari *function* dalam tipe data DATETIME sesuai dengan tipe data nilai variabel pada tabel `mem_datetime_variables`.
2. Tabel yang diakses dalam *function* ini ialah tabel dengan memory engine.
3. Masukan *function* ialah nama variabel.

4.3.1.4 Implementasi Function `get_datetime_variables_from_id`

Function yang digunakan untuk mendapatkan nilai dari variabel dengan tipe data datetime pada tabel `mem_datetime_variables`.

Tabel 4.7 Pseudocode Function `get_datetime_variables_from_id`

Pseudocode Function <code>get_datetime_variables_from_id</code>	
Deklarasi awal	
var_iddv:INT -> Masukan dari user/sistem	
var_value:DATETIME	
Deskripsi	
1	MULAI
2	
3	SELECT value INTO var_value
4	FROM mem_datetime_variables
5	WHERE iddv = var_iddv;
6	
7	RETURN var_value;
8	
9	SELESAI

Penjelasan dari *function* `get_datetime_variables_from_id` pada tabel 4.7 yaitu:

1. Keluaran dari *function* dalam tipe data DATETIME sesuai dengan tipe data nilai variabel pada tabel mem_datetime_variables.
2. Tabel yang diakses dalam *function* ini ialah tabel dengan memory engine.
3. Masukan *function* ialah ID variabel.

4.3.1.5 Implementasi Function get_datetime_variables_id

Function yang digunakan untuk mendapatkan ID dari variabel pada tabel mem_datetime_variables.

Tabel 4.8 Pseudocode Function get_datetime_variables_id

Pseudocode Function get_datetime_variables_id	
Deklarasi awal	
var_variable:VARCHAR(45) -> Masukan dari user/sistem	
var_iddv:INT	
Deskripsi	
1	MULAI
2	
3	SELECT iddv INTO var_iddv
4	FROM mem_datetime_variables
5	WHERE variable = var_variable;
6	
7	RETURN var_iddv;
8	
9	SELESAI

Penjelasan dari *function* get_datetime_variables_id pada tabel 4.8 yaitu:

1. Keluaran dari *function* dalam tipe data INT sesuai dengan tipe data ID variabel pada tabel mem_datetime_variables.
2. Tabel yang diakses dalam *function* ini ialah tabel dengan memory engine.
3. Masukan *function* ialah nama variabel.

4.3.1.6 Implementasi Function `get_distance`

Function yang digunakan untuk mendapatkan jarak antara 2 desa yakni desa pemain dengan desa tujuan serang pemain. Pencarian jarak dengan menerapkan rumus *phytagoras* pada koordinat kedua desa.

Tabel 4.9 Pseudocode Function `get_distance`

Pseudocode Function <code>get_distance</code>	
Deklarasi awal	
<pre> var_village:INT -> Masukan dari user/sistem var_target_village:INT -> Masukan dari user/sistem var_village_x:INT var_village_y:INT var_target_village_x:INT var_target_village_y:INT var_distance:FLOAT </pre>	
Deskripsi	
1	MULAI
2	
3	CALL <code>get_village_location</code> (<code>var_village</code> , <code>var_village_x</code> ,
4	<code>var_village_y</code>);
5	
6	CALL
7	<code>get_village_location</code> (<code>var_target_village</code> , <code>var_target</code>
8	<code>village_x</code> , <code>var_target_village_y</code>);
9	
10	SET <code>var_distance</code> = $\text{SQRT}(\text{POW}(\text{var\_target\_village\_x}$
11	$\text{var\_village\_x},2)+\text{POW}(\text{var\_target\_village\_y}$
12	$\text{var\_village\_y},2))$;
13	
14	RETURN <code>var_distance</code> ;
15	
16	SELESAI

Penjelasan dari *function* `get_distance` pada tabel 4.9 yaitu:

1. Keluaran dari *function* dalam tipe data FLOAT.

2. *Function* SQRT pada baris 10 digunakan untuk menghasilkan akar pangkat 2 dari suatu bilangan.
3. *Function* POW pada bari 10-11 berfungsi untuk menghasilkan pangkat 2 atau kuadrat dari suatu bilangan.

4.3.1.7 Implementasi Function get_integer_variables

Function yang digunakan untuk mendapatkan nilai dari variabel dengan tipe data integer pada tabel mem_integer_variables.

Tabel 4.10 Pseudocode Function get_integer_variables

Pseudocode Function get_integer_variables	
Deklarasi awal	
var_variable:VARCHAR(45) -> Masukan dari user/sistem	
var_value:INT	
Deskripsi	
1	MULAI
2	
3	SELECT value INTO var_value
4	FROM mem_integer_variables
5	WHERE variable = var_variable;
6	
7	RETURN var_value;
8	
9	SELESAI

Penjelasan dari *function* get_integer_variables pada tabel 4.10 yaitu:

1. Keluaran dari *function* dalam tipe data INT sesuai dengan tipe data nilai variabel pada tabel mem_integer_variables.
2. Tabel yang diakses dalam *function* ini ialah tabel dengan memory engine.
3. Masukan *function* ialah nama variabel.

4.3.1.8 Implementasi Function `get_integer_variables_from_id`

Function yang digunakan untuk mendapatkan nilai dari variabel dengan tipe data integer pada tabel `mem_integer_variables`.

Tabel 4.11 Pseudocode Function `get_integer_variables_from_id`

Pseudocode Function <code>get_integer_variables_from_id</code>	
Deklarasi awal	
var_idiv:INT -> Masukan dari user/sistem	
var_value:INT	
Deskripsi	
1	MULAI
2	
3	SELECT value INTO var_value
4	FROM mem_integer_variables
5	WHERE idiv = var_idiv;
6	
7	RETURN var_value;
8	
9	SELESAI

Penjelasan dari *function* `get_integer_variables_from_id` pada tabel 4.11 yaitu:

1. Keluaran dari *function* dalam tipe data INT sesuai dengan tipe data nilai variabel pada tabel `mem_integer_variables`.
2. Tabel yang diakses dalam *function* ini ialah tabel dengan memory engine.
3. Masukan *function* ialah ID variabel.

4.3.1.9 Implementasi Function `get_integer_variables_id`

Function yang digunakan untuk mendapatkan ID dari variabel pada tabel `mem_integer_variables`.

Tabel 4.12 Pseudocode Function `get_integer_variables_id`

Pseudocode Function <code>get_integer_variables_id</code>	
Deklarasi awal <code>var_variable:VARCHAR(45) -> Masukan dari user/sistem</code> <code>var_idiv:INT</code> Deskripsi	
1	MULAI
2	
3	<code>SELECT idiv INTO var_idiv</code>
4	<code>FROM mem_integer_variables</code>
5	<code>WHERE variable = var_variable;</code>
6	
7	<code>RETURN var_idiv;</code>
8	
9	SELESAI

Penjelasan dari *function* `get_integer_variables_id` pada tabel 4.12 yaitu:

1. Keluaran dari *function* dalam tipe data INT sesuai dengan tipe data ID variabel pada tabel `mem_integer_variables`.
2. Tabel yang diakses dalam *function* ini ialah tabel dengan memory engine.
3. Masukan *function* ialah nama variabel.

4.3.1.10 Implementasi Function `get_new_attack`

Function yang digunakan untuk mendapatkan ID penyerangan atau *attack* baru yang nantinya akan dimasukkan pada tabel `villages_troops attacks`.

Tabel 4.13 Pseudocode Function `get_new_attack`

Pseudocode Function <code>get_new_attack</code>
Deklarasi awal <code>var attack:INT</code>

Deskripsi	
1	MULAI
2	
3	SELECT MAX(attack)
4	INTO var_attack
5	FROM villages_troops_attacks LOCK IN SHARE MODE;
6	
7	IF var_attack IS NULL
8	THEN
9	SET var_attack = 0;
10	ELSE
11	SET var_attack = var_attack;
12	END IF;
13	
14	SET var_attack = var_attack + 1;
15	
16	RETURN var_attack;
17	
18	SELESAI

Penjelasan dari *function* `get_new_attack` pada tabel 4.13 yaitu:

1. Keluaran dari *function* dalam tipe data INT.
2. *Function* MAX pada baris 3 berfungsi untuk mencari nilai tertinggi pada kolom tabel.
3. Seleksi kondisi pada baris 7-12 dipakai untuk memberikan nilai 0 pada variabel `var_attack` karena nilai NULL tidak bisa dikenakan operasi penjumlahan pada baris 14.

4.3.1.11 Implementasi Function `get_new_idv`

Function yang digunakan untuk mendapatkan ID desa baru yang nantinya akan dimasukkan pada tabel `villages`. Penggunaan *function* ialah untuk menggantikan fitur AUTO INCREMENT.

Tabel 4.14 Pseudocode Function `get_new_idv`

Pseudocode Function <code>get_new_idv</code>	
Deklarasi awal	
<code>var_idv:INT</code>	
Deskripsi	
1	MULAI
2	
3	<code>SELECT MAX(idv)</code>
4	<code>INTO var_idv</code>
5	<code>FROM villages LOCK IN SHARE MODE;</code>
6	
7	<code>IF var_idv IS NULL</code>
8	<code>THEN</code>
9	<code>SET var_idv = 0;</code>
10	<code>ELSE</code>
11	<code>SET var_idv = var_idv;</code>
12	<code>END IF;</code>
13	
14	<code>SET var_idv = var_idv + 1;</code>
15	
16	<code>RETURN var_idv;</code>
17	
18	SELESAI

Penjelasan dari *function* `get_new_idv` pada tabel 4.14 yaitu:

1. Keluaran dari *function* dalam tipe data INT.
2. *Function* MAX pada baris 3 berfungsi untuk mencari nilai tertinggi pada kolom tabel.
3. Seleksi kondisi pada baris 7-12 dipakai untuk memberikan nilai 0 pada variabel `var_idv` karena nilai NULL tidak bisa dikenakan operasi penjumlahan pada baris 14.

4.3.1.12 Implementasi Function `get_new_idvas`

Function yang digunakan untuk mendapatkan ID penjadwalan sumber daya baru yang akan dimasukkan pada tabel `villages_resources_schedules`. Penggunaan *function* ialah untuk menggantikan fitur AUTO INCREMENT.

Tabel 4.15 Pseudocode Function `get_new_idvas`

Pseudocode Function <code>get_new_idvas</code>	
Deklarasi awal	
<code>var_idvas:INT</code>	
Deskripsi	
1	MULAI
2	
3	<code>SELECT MAX(idvas)</code>
4	<code>INTO var_idvas</code>
5	<code>FROM villages_resources_schedules LOCK IN SHARE MODE;</code>
6	
7	<code>IF var_idvas IS NULL</code>
8	<code>THEN</code>
9	<code>SET var_idvas = 0;</code>
10	<code>ELSE</code>
11	<code>SET var_idvas = var_idvas;</code>
12	<code>END IF;</code>
13	
14	<code>SET var_idvas = var_idvas + 1;</code>
15	
16	<code>RETURN var_idvas;</code>
17	
18	SELESAI

Penjelasan dari *function* `get_new_idvas` pada tabel 4.15 yaitu:

1. Keluaran dari *function* dalam tipe data INT.
2. *Function* MAX pada baris 3 berfungsi untuk mencari nilai tertinggi pada kolom tabel.

3. Seleksi kondisi pada baris 7-12 dipakai untuk memberikan nilai 0 pada variabel `var_idvas` karena nilai NULL tidak bisa dikenakan operasi penjumlahan pada baris 14.

4.3.1.13 Implementasi Function `get_new_idvt`

Function yang digunakan untuk mendapatkan ID pasukan pertahanan desa maupun pasukan penyerang baru yang nantinya akan dimasukkan pada tabel `villages_troops`. Penggunaan *function* ialah untuk menggantikan fitur AUTO INCREMENT.

Tabel 4.16 Pseudocode Function `get_new_idvt`

Pseudocode Function <code>get_new_idvt</code>	
Deklarasi awal	
<code>var_idvt:INT</code>	
Deskripsi	
1	MULAI
2	
3	<code>SELECT MAX(idvt)</code>
4	<code>INTO var_idvt</code>
5	<code>FROM villages_troops LOCK IN SHARE MODE;</code>
6	
7	<code>IF var_idvt IS NULL</code>
8	<code>THEN</code>
9	<code>SET var_idvt = 0;</code>
10	<code>ELSE</code>
11	<code>SET var_idvt = var_idvt;</code>
12	<code>END IF;</code>
13	
14	<code>SET var_idvt = var_idvt + 1;</code>
15	
16	<code>RETURN var_idvt;</code>
17	
18	SELESAI

Penjelasan dari *function* `get_new_idv` pada tabel 4.16 yaitu:

1. Keluaran dari *function* dalam tipe data INT.
2. *Function* MAX pada baris 3 berfungsi untuk mencari nilai tertinggi pada kolom tabel.
3. Seleksi kondisi pada baris 7-12 dipakai untuk memberikan nilai 0 pada variabel `var_idvt` karena nilai NULL tidak bisa dikenakan operasi penjumlahan pada baris 14.

4.3.1.14 Implementasi Function `get_travel_time`

Function yang digunakan untuk mendapatkan waktu tempuh antara 2 desa yakni desa pemain dengan desa tujuan serang pemain berdasarkan kecepatan pasukan yang dipilih. Pencarian jarak dengan menerapkan rumus waktu tempuh yakni jarak dibagi kecepatan.

Tabel 4.17 Pseudocode Function `get_travel_time`

Pseudocode Function <code>get_travel_time</code>	
Deklarasi awal <pre> var_troop:INT -> Masukan dari user/sistem var_village:INT -> Masukan dari user/sistem var_target_village:INT -> Masukan dari user/sistem var_travel_time:TIME var_distance:INT var_speed:INT </pre> Deskripsi	
1	MULAI
2	
3	SET <code>var_distance =</code>
4	<code>get_distance(var_village,var_target_village);</code>
5	
6	SELECT <code>iv.value INTO var_speed</code>
7	FROM <code>troops_stats ts</code>
8	LEFT JOIN <code>mem_integer_variables iv ON ts.value_iv =</code>
9	<code>iv.idiv</code>

10	LEFT JOIN attributes a ON ts.attribute = a.ida
11	WHERE a.attribute = "troop_speed"
12	AND ts.troop = var_troop;
13	
14	SET var_travel_time := SEC_TO_TIME(var_distance /
15	var_speed * 3600);
16	
17	RETURN var_travel_time;
18	
19	SELESAI

Penjelasan dari *function* get_travel_time pada tabel 4.17 yaitu:

1. Keluaran dari *function* dalam tipe data TIME.
2. Dalam melakukan pencarian jarak antara 2 desa digunakan *function* get_distance.
3. Perkalian hasil dengan 3600 pada baris 14-15 karena kecepatan pasukan dalam satuan jarak perjam dan hasil yang diperlukan harus dalam satuan perdetik.

4.3.1.15 Implementasi Function get_troop_id

Function yang digunakan untuk mendapatkan ID dari pasukan yang diperlukan ketika pemrosesan.

Tabel 4.18 Pseudocode Function get_troop_id

Pseudocode Function get_troop_id	
Deklarasi awal	
var_troop:VARCHAR(45) -> Masukan dari user/sistem	
var_idt:INT	
Deskripsi	
1	MULAI
2	
3	SELECT idt INTO var_idt
4	FROM troops
5	WHERE troop = var_troop;

6	
7	RETURN var_idt;
8	
9	SELESAI

Penjelasan dari *function* get_troop_id pada tabel 4.18 yaitu:

1. Keluaran dari *function* dalam tipe data INT sesuai dengan tipe data ID pasukan pada tabel troops.
2. Masukan *function* ialah nama pasukan seperti bhayangkara dan kris_swordman.

4.3.1.16 Implementasi Function get_varchar_variables

Function yang digunakan untuk mendapatkan nilai dari variabel dengan tipe data varchar pada tabel mem_varchar_variables.

Tabel 4.19 Pseudocode Function get_varchar_variables

Pseudocode Function get_varchar_variables	
Deklarasi awal	
var_variable:VARCHAR(45) --> Masukan dari user/sistem	
var_value:VARCHAR(45)	
Deskripsi	
1	MULAI
2	
3	SELECT value INTO var_value
4	FROM mem_varchar_variables
5	WHERE variable = var_variable;
6	
7	RETURN var_value;
8	
9	SELESAI

Penjelasan dari *function* get_varchar_variables pada tabel 4.19 yaitu:

1. Keluaran dari *function* dalam tipe data VARCHAR sesuai dengan tipe data nilai variabel pada tabel mem_varchar_variables.
2. Tabel yang diakses dalam *function* ini ialah tabel dengan memory engine.
3. Masukan *function* ialah nama variabel.

4.3.1.17 Implementasi Function get_varchar_variables_from_id

Function yang digunakan untuk mendapatkan nilai dari variabel dengan tipe data varchar pada tabel mem_varchar_variables.

Tabel 4.20 Pseudocode Function get_varchar_variables_from_id

Pseudocode Function get_varchar_variables_from_id	
Deklarasi awal	
var_idvv:INT -> Masukan dari user/sistem	
var_value:VARCHAR(45)	
Deskripsi	
1	MULAI
2	
3	SELECT value INTO var_value
4	FROM mem_varchar_variables
5	WHERE idvv = var_idvv;
6	
7	RETURN var_value;
8	
9	SELESAI

Penjelasan dari *function* get_varchar_variables_from_id pada tabel 4.20 yaitu:

1. Keluaran dari *function* dalam tipe data VARCHAR sesuai dengan tipe data nilai variabel pada tabel mem_varchar_variables.
2. Tabel yang diakses dalam *function* ini ialah tabel dengan memory engine.
3. Masukan *function* ialah ID variabel.

4.3.1.18 Implementasi Function `get_varchar_variables_id`

Function yang digunakan untuk mendapatkan ID dari variabel pada tabel `mem_varchar_variables`.

Tabel 4.21 Pseudocode Function `get_varchar_variables_id`

Pseudocode Function <code>get_varchar_variables_id</code>	
Deklarasi awal	
<code>var_variable:VARCHAR(45) -> Masukan dari user/sistem</code> <code>var_idvv:INT</code>	
Deskripsi	
1	MULAI
2	
3	<code>SELECT idvv INTO var_idvv</code>
4	<code>FROM mem_varchar_variables</code>
5	<code>WHERE variable = var_variable;</code>
6	
7	<code>RETURN var_idvv;</code>
8	
9	SELESAI

Penjelasan dari *function* `get_varchar_variables_id` pada tabel 4.21 yaitu:

1. Keluaran dari *function* dalam tipe data INT sesuai dengan tipe data ID variabel pada tabel `mem_varchar_variables`.
2. Tabel yang diakses dalam *function* ini ialah tabel dengan memory engine.
3. Masukan *function* ialah nama variabel.

4.3.1.19 Implementasi Function `get_village_axis`

Function yang digunakan untuk mendapatkan posisi koordinat sumbu x atau sumbu y dari desa para pemain. Dalam sistem atribut, sumbu x disebut `x_axis` dan `y_axis` untuk atribut sumbu y.

Tabel 4.22 Pseudocode Function get_village_axis

Pseudocode Function get_village_axis	
Deklarasi awal	
var_village:INT -> Masukan dari user/sistem var_axis:VARCHAR(50) -> Masukan dari user/sistem var_attribute:INT var_value:INT	
Deskripsi	
1	MULAI
2	
3	SET var_attribute = get_attribute_id(var_axis);
4	
5	SELECT value INTO var_value
6	FROM villages_locations
7	WHERE village = var_village AND attribute =
8	var_attribute;
9	
10	RETURN var_value;
11	
12	SELESAI

Penjelasan dari *function* get_village_axis pada tabel 4.22 yaitu:

1. Keluaran dari *function* dalam tipe data INT sesuai dengan tipe data nilai atribut pada tabel villages_locations.
2. Atribut yang bisa dipilih hanya x_axis atau y_axis.

4.3.1.20 Implementasi Function random_integer

Function yang digunakan untuk mendapatkan nilai acak pada rentan nilai tertentu dalam tipe data integer.

Tabel 4.23 Pseudocode Function random_integer

Pseudocode Function random_integer	

Deklarasi awal	
<pre> var_min_value:INT -> Masukan dari user/sistem var_max_value:INT -> Masukan dari user/sistem var_value:INT </pre>	
Deskripsi	
1	MULAI
2	
3	SET var_value = FLOOR(var_min_value + (RAND() * (var_max_value + 1 - var_min_value)));
4	
5	
6	RETURN var_value;
7	
8	SELESAI

Penjelasan dari *function* random_integer pada tabel 4.23 yaitu:

1. Keluaran dari *function* dalam tipe data INT.
2. Masukan berupa batas bawah dan batas atas nilai acak yang diinginkan.
3. *Function* FLOOR pada baris 3 digunakan untuk membulatkan nilai acak hasil *generate function* RAND pada baris 3-4.

4.3.2 Procedure

Ragam algoritma *procedure* dalam *game server online* permainan *strategy* Kerajaan Majapahit ialah sebagai berikut.

4.3.2.1 Implementasi Procedure build_village

Salah satu bagian *procedure* dari proses pendaftaran pemain yang dipakai untuk membangun desa pemain. Pemakaian hanya berlangsung sekali saat pemain melakukan pendaftaran.

Tabel 4.24 Pseudocode Procedure build_village

Pseudocode Procedure build_village
Deklarasi awal

```
var_player:INT -> Masukan dari user/sistem
var_village_name:VARCHAR(50) -> Masukan dari user/sistem
var_x_axis:INT -> Masukan dari user/sistem
var_y_axis:INT -> Masukan dari user/sistem
```

Deskripsi

1	MULAI
2	
3	INSERT INTO temp_villages
4	SET
5	player = var_player,
6	village_name = var_village_name,
7	x_axis = var_x_axis,
8	y_axis = var_y_axis;
9	
10	SELESAI

Penjelasan dari *procedure* build_village pada tabel 4.24 yaitu:

1. Nilai masukan dari var_player di *generate* oleh sistem melalui *function* get_new_idv.
2. Nilai masukan var_x_axis dan var_y_axis didapatkan dari *generate* dengan *procedure* random_village_axis.
3. Nilai-nilai masukan ke tabel temp_villages yang memakai *memory engine* bersifat sementara dan nantinya akan diproses *trigger* yang ada dalam tabel.

4.3.2.2 Implementasi Procedure buy_building

Salah satu bagian *procedure* yang dipakai oleh pemain untuk membangun suatu bangunan di desa pemain. Fungsi dasarnya ialah untuk mengurangi sumber daya pemain manakala pemain membeli bangunan serta menjadwal pembangunan.

Tabel 4.25 Pseudocode Procedure buy_building

Pseudocode Procedure buy_building

Deklarasi awal

```
var_village:INT -> Masukan dari user/sistem
var_building:INT -> Masukan dari user/sistem
var_idbp:INT
var_building_id:INT
var_attribute_id:INT
var_value:INT
var_execution_time:TIMESTAMP = NOW()
var_finished_time:TIMESTAMP
var_status_id:INT
var_finished:INT = 0
var_verification:INT
curs:CURSOR FOR
    SELECT bp.idbp,bp.building,bp.attribute,iv.value
    FROM buildings_prices bp
    LEFT JOIN mem_integer_variables iv ON bp.value_iv =
    iv.idiv
    WHERE building = var_building
CONTINUE HANDLER FOR NOT FOUND SET var_finished = 1
```

Deskripsi

1	MULAI
2	
3	OPEN curs;
4	sync_process: LOOP
5	
6	FETCH curs INTO
7	var_idbp,var_building_id,var_attribute_id,var_value;
8	
9	IF var_finished = 1 THEN
10	LEAVE sync_process;
11	END IF;
12	
13	INSERT INTO villages_resources
14	SET
15	village = var_village,
16	attribute = var_attribute_id,
17	value = (var_value * -1),
18	execution_time = var_execution_time;
19	
20	END LOOP sync_process;
21	
22	CLOSE curs;
23	

```

24 SELECT (NOW() + dv.value) INTO var_finished_time
25 FROM buildings_build_times bbt
26 LEFT JOIN mem_datetime_variables dv ON dv.iddv =
27 bbt.build_time_dv
28 WHERE bbt.building = var_building;
29
30 SET var_status_id =
31 get_varchar_variables_id("iteration_once");
32
33 INSERT INTO villages_buildings_builds_schedules
34 SET
35 village = var_village,
36 building = var_building,
37 finished = var_finished_time,
38 status_vv = var_status_id;
39
40 SELESAI

```

Penjelasan dari *procedure* buy_building pada tabel 4.25 yaitu:

1. *Function* NOW pada deklarasi awal var_execution_time berfungsi untuk mendapatkan waktu sistem tatkala *function* dipanggil.
2. Deklarasi awal curs digunakan untuk mendapatkan harga dari bangunan yang kemudian pada perulangan di baris 4-20 harga bangunan tersebut digunakan untuk mengurangi sumber daya di desa pemain.
3. Baris 24-28 digunakan untuk mendapatkan waktu selesai pembangunan yang nantinya dimasukkan di penjadwalan pembangunan bangunan pada tabel villages_buildings_builds_schedules.

4.3.2.3 Implementasi Procedure buy_troop

Salah satu bagian *procedure* yang dipakai oleh pemain untuk melatih suatu pasukan di desa pemain. Fungsi dasarnya ialah untuk mengurangi sumber daya pemain manakala pemain membeli pasukan serta menjadwal pelatihan.

Tabel 4.26 Pseudocode Procedure buy_troop

Pseudocode Procedure buy_troop

Deklarasi awal

```
var_village:INT -> Masukan dari user/sistem
var_troop:INT -> Masukan dari user/sistem
var_troop_amount:INT -> Masukan dari user/sistem
var_idtp:INT
var_troop_id:INT
var_attribute_id:INT
var_value:INT
var_execution_time:TIMESTAMP = NOW()
var_finished_time:TIMESTAMP
var_status_id:INT
var_finished:INT = 0
var_verification:INT
curs:CURSOR FOR
    SELECT tp.idtp,tp.troop,tp.attribute,iv.value
    FROM troops_prices tp
    LEFT JOIN mem_integer_variables iv ON tp.value_iv
    = iv.idiv
    WHERE troop = var_troop
CONTINUE HANDLER FOR NOT FOUND SET var_finished = 1
```

Deskripsi

1	MULAI
2	
3	OPEN curs;
4	sync_process: LOOP
5	
6	FETCH curs INTO
7	var_idtp,var_troop_id,var_attribute_id,var_value;
8	
9	IF var_finished = 1 THEN
10	LEAVE sync_process;
11	END IF;
12	
13	INSERT INTO villages_resources
14	SET
15	village = var_village,
16	attribute = var_attribute_id,
17	value = (var_value * var_troop_amount * -1),
18	execution_time = var_execution_time;
19	
20	END LOOP sync_process;
21	
22	CLOSE curs;


```

23
24 SELECT (NOW() + dv.value) INTO var_finished_time
25 FROM troops_build_times tbt
26 LEFT JOIN mem_datetime_variables dv ON dv.iddv =
27 tbt.build_time_dv
28 WHERE tbt.troop = var_troop;
29
30 SET var_status_id =
31 get_varchar_variables_id("iteration_once");
32
33 INSERT INTO villages_troops_builds_schedules
34 SET
35 village = var_village,
36 troop = var_troop,
37 value = var_troop_amount,
38 finished = var_finished_time,
39 status_vv = var_status_id;
40
41 SELESAI

```

Penjelasan dari *procedure* buy_troop pada tabel 4.26 yaitu:

1. *Function* NOW pada deklarasi awal var_execution_time berfungsi untuk mendapatkan waktu sistem tatkala *function* dipanggil.
2. Deklarasi awal curs digunakan untuk mendapatkan harga dari pasukan yang kemudian pada perulangan di baris 4-20 harga bangunan tersebut digunakan untuk mengurangi sumber daya di desa pemain.
3. Baris 24-28 digunakan untuk mendapatkan waktu selesai pembangunan yang nantinya dimasukkan di penjadwalan pelatihan pasukan pada tabel villages_troops_builds_schedules.

4.3.2.4 Implementasi Procedure default_village

Salah satu bagian *procedure* dari proses pendaftaran pemain yang dipakai untuk melakukan *generate* nilai *default* untuk atribut desa pemain kemudian meneruskan untuk diproses di *procedure* selanjutnya. Pemakaian hanya berlangsung sekali saat pemain melakukan pendaftaran.

Tabel 4.27 Pseudocode Procedure default_village

Pseudocode Procedure default_village	
Deklarasi awal	
var_player:INT -> Masukan dari user/sistem var_username:VARCHAR(45) -> Masukan dari user/sistem var_x_axis:INT var_y_axis:INT var_village_name:VARCHAR(45)	
Deskripsi	
1	MULAI
2	
3	CALL random_village_axis(var_x_axis,var_y_axis);
4	
5	SET var_village_name = CONCAT(var_username,"'s
6	village");
7	
8	CALL build_village(var_player,var_village_name,
9	var_x_axis,var_y_axis);
10	
11	SELESAI

Penjelasan dari *procedure* default_village pada tabel 4.27 yaitu:

1. Atribut desa yang di-generate ialah x_axis dan y_axis.
2. Nama desa didapatkan dengan *function* CONCAT MySQL yang berfungsi untuk menggabungkan username pemain dengan “'s village”.
3. Nilai-nilai hasil generate di *procedure* ini dilanjutkan untuk diproses di *procedure* build_village yang ditunjukkan di baris 8-9.
4. Procedure akan digunakan dalam *trigger* pada tabel players.

4.3.2.5 Implementasi Procedure default_village_buildings

Salah satu bagian *procedure* dari proses pendaftaran pemain yang dipakai untuk melakukan *generate* bangunan *default* untuk desa pemain. Pemakaian hanya berlangsung sekali saat pemain melakukan pendaftaran.

Tabel 4.28 Pseudocode Procedure default_village_buildings

Pseudocode Procedure default_village_buildings	
Deklarasi awal	
var_village:INT -> Masukan dari user/sistem	
var_building_id:INT	
Deskripsi	
1	MULAI
2	
3	SET var_building_id = get_building_id("town_center");
4	CALL
5	village_building_registration(var_village,var_building_id,1);
6	
7	SET var_building_id = get_building_id("farm");
8	CALL
9	village_building_registration(var_village,var_building_id,1);
10	
11	SET var_building_id = get_building_id("lumber_camp");
12	CALL
13	village_building_registration(var_village,var_building_id,1);
14	
15	SET var_building_id = get_building_id("mining_camp");
16	CALL
17	village_building_registration(var_village,var_building_id,1);
18	
19	SELESAI

Penjelasan dari *procedure* default_village_buildings pada tabel 4.28 yaitu:

1. Bangunan yang termasuk dalam bangunan *default* ialah farm, lumber_camp dan mining_camp.
2. Setiap desa hanya mendapatkan 1 bangunan untuk tiap jenis bangunan *default* yang diberikan.

4.3.2.6 Implementasi Procedure default_village_resources

Salah satu bagian *procedure* dari proses pendaftaran pemain yang dipakai untuk melakukan pemberian sejumlah sumber daya untuk desa pemain di awal permainan. Pemakaian hanya berlangsung sekali saat pemain melakukan pendaftaran.

Tabel 4.29 Pseudocode Procedure default_village_resources

Pseudocode Procedure default_village_resources	
Deklarasi awal	
var_village:INT -> Masukan dari user/sistem var_x_axis:INT -> Masukan dari user/sistem var_y_axis:INT -> Masukan dari user/sistem var_execution_time:TIMESTAMP -> Masukan dari user/sistem var_default_food:INT var_default_wood:INT var_default_gold:INT	
Deskripsi	
1	MULAI
2	
3	SET var_default_food =
4	get_integer_variables("default_food");
5	SET var_default_wood =
6	get_integer_variables("default_wood");
7	SET var_default_gold =
8	get_integer_variables("default_gold");
9	
10	CALL
11	insert_village_location(var_village,"x_axis",var_x_axis);
12	CALL
13	insert_village_location(var_village,"y_axis",var_y_axis);
14	CALL insert_village_resource(var_village,"food",
15	var_default_food,var_execution_time);
16	CALL insert_village_resource(var_village,"wood",
17	var_default_wood,var_execution_time);
18	CALL insert_village_resource(var_village,"gold",
19	var_default_gold,var_execution_time);
20	
21	SELESAI

Penjelasan dari *procedure* default_village_troops pada tabel 4.29 yaitu:

1. Semua jenis sumber daya akan diberikan dalam jumlah tertentu pada pemain saat mendaftar permainan.
2. Jumlah dari tiap jenis sumber daya yang diberikan didapat dari tabel integer_variables.

4.3.2.7 Implementasi Procedure generate_village_resource_schedule_from_resources_buildings

Bangunan penghasil sumber daya berfungsi untuk menghasilkan sumber daya secara rutin dan terjadwal dalam tiap kurun waktu tertentu. *Procedure* ini berfungsi untuk *generate* jadwal penghasilan sumber daya tiap dilakukannya pembangunan bangunan penghasil sumber daya.

Tabel 4.30 Pseudocode Procedure generate_village_resource_schedule_from_resources_buildings

Pseudocode Procedure generate_village_resource_schedule_from_resources_buildings	
Deklarasi awal	
var_idvb:INT -> Masukan dari user/sistem var_idv:INT -> Masukan dari user/sistem var_idb:INT -> Masukan dari user/sistem var_attribute_id:INT var_value_id:INT var_interlude_id:INT var_iteration_id:INT var_execution_time:TIMESTAMP var_idvas:INT	
Deskripsi	
1	MULAI
2	
3	IF(EXISTS(SELECT idrb FROM resources_buildings
4	WHERE building = var idb) = TRUE)

```

5 THEN
6     SELECT attribute,value_iv,interlude_dv,
7         iteration_vv
8     INTO
9     var_attribute_id,var_value_id,var_interlude_id,
10    var_iteration_id
11    FROM resources_buildings WHERE building =
12    var_idb;
13
14    SET var_execution_time = NOW() +
15    get_datetime_variables_from_id(var_interlude_id);
16
17    SET var_idvas = get_new_idvas();
18
19    INSERT
20    INTO villages_resources_schedules SET
21    idvas = var_idvas,
22    village = var_idv,
23    attribute = var_attribute_id,
24    value_iv = var_value_id,
25    execution_time = var_execution_time,
26    interlude_dv = var_interlude_id,
27    iteration_vv = var_iteration_id;
28
29    INSERT INTO
30    villages_buildings_resources_schedules SET
31    village_building = var_idvb,
32    village_resource_schedule = var_idvas;
33 END IF;
34
35 SELESAI

```

Penjelasan dari *procedure* generate_village_resource_schedule_from_resources_buildings pada tabel 4.30 yaitu:

1. Procedure ditempatkan pada *trigger* di tabel *villages_buildings*.
2. Prinsip dasar kerja *procedure* ialah melakukan cek terhadap bangunan yang dibangun. Apabila termasuk bangunan penghasil sumber daya maka akan dibuatkan jadwal penghasilan sumber daya.
3. Algoritma pengecekan terletak pada baris 3-4.
4. Procedure akan digunakan dalam *trigger* pada tabel *villages_buildings*.

4.3.2.8 Implementasi Procedure `get_village_location`

Procedure yang digunakan untuk mendapatkan posisi koordinat sumbu x dan sumbu y dari desa para pemain. Dalam sistem atribut, sumbu x disebut `x_axis` dan `y_axis` untuk atribut sumbu y.

Tabel 4.31 Pseudocode Procedure `get_village_location`

Pseudocode Procedure <code>get_village_location</code>	
Deklarasi awal	
<code>var_village:INT -> Masukan dari user/sistem</code> <code>var_x_axis:INT -> Keluaran dari sistem</code> <code>var_y_axis:INT -> Keluaran dari sistem</code>	
Deskripsi	
1	MULAI
2	
3	SET <code>var_x_axis =</code>
4	<code>get_village_axis(var_village,"x_axis");</code>
5	
6	SET <code>var_y_axis =</code>
7	<code>get_village_axis(var_village,"y_axis");</code>
8	
9	SELESAI

Penjelasan dari *procedure* `get_village_location` pada tabel 4.31 yaitu:

1. Keluaran dari *procedure* dalam tipe data INT sesuai dengan tipe data nilai atribut pada tabel `villages_locations`.
2. Kedua atribut lokasi desa pemain yang dapat dihasilkan yakni `x_axis` dan `y_axis`.

4.3.2.9 Implementasi Procedure `insert_building_build_time`

Procedure yang digunakan untuk memasukkan data waktu pembangunan tiap bangunan dalam permainan. Data nantinya akan dimasukkan dalam tabel `buildings_build_times`.

Tabel 4.32 Pseudocode Procedure insert_building_build_time

Pseudocode Procedure insert_building_build_time	
Deklarasi awal	
var_building:VARCHAR(50) -> Masukan dari user/sistem	
var_build_time:VARCHAR(50) -> Masukan dari user/sistem	
var_building_id:INT	
var_datetime_id:INT	
Deskripsi	
1	MULAI
2	
3	SET var_building_id = get_building_id(var_building);
4	
5	SET var_datetime_id =
6	get_datetime_variables_id(var_build_time);
7	
8	INSERT INTO buildings_build_times
9	SET
10	building := var_building_id,
11	build_time_dv := var_datetime_id;
12	
13	SELESAI

Penjelasan dari *procedure* insert_building_build_time pada tabel 4.32 yaitu:

1. Masukan dari *procedure* dalam tipe data VARCHAR.
2. Masukan berupa nama bangunan dan nama variabel DATETIME yang mewakili waktu pembangunan.
3. Nama bangunan dan nama variabel nantinya akan dicari ID-nya untuk dimasukkan dalam tabel buildings_build_times.

4.3.2.10 Implementasi Procedure insert_building_price

Procedure yang digunakan untuk memasukkan data harga tiap bangunan dalam permainan. Data nantinya akan dimasukkan dalam tabel buildings_prices.

Tabel 4.33 Pseudocode Procedure insert_building_price

Pseudocode Procedure insert_building_price	
Deklarasi awal	
var_building:VARCHAR(50) -> Masukan dari user/sistem var_attribute:VARCHAR(50) -> Masukan dari user/sistem var_value:VARCHAR(50) -> Masukan dari user/sistem var_building_id:INT var_attribute_id:INT var_value_id:INT	
Deskripsi	
1	MULAI
2	
3	SET var_building_id = get_building_id(var_building);
4	
5	SET var_attribute_id =
6	get_attribute_id(var_attribute);
7	
8	SET var_value_id =
9	get_integer_variables_id(var_value);
10	
11	INSERT INTO buildings_prices
12	SET
13	building = var_building_id,
14	attribute = var_attribute_id,
15	value_iv = var_value_id;
16	
17	SELESAI

Penjelasan dari *procedure* insert_building_price pada tabel 4.33 yaitu:

1. Masukan dari *procedure* dalam tipe data VARCHAR.
2. Masukan berupa nama bangunan, nama jenis sumber daya dan nama variabel INTEGER yang mewakili harga bangunan.
3. Nama bangunan, nama jenis sumber daya dan nama variabel nantinya akan dicari ID-nya untuk dimasukkan dalam tabel buildings_prices.

4.3.2.11 Implementasi Procedure insert_resource_building

Procedure yang digunakan untuk memasukkan data bangunan dalam permainan yang termasuk bangunan penghasil sumber daya. Data nantinya akan dimasukkan dalam tabel resources_buildings.

Tabel 4.34 Pseudocode Procedure insert_resource_building

Pseudocode Procedure insert_resource_building	
Deklarasi awal	
var_building:VARCHAR(45) -> Masukan dari user/sistem var_attribute:VARCHAR(45) -> Masukan dari user/sistem var_value:VARCHAR(45) -> Masukan dari user/sistem var_interlude:VARCHAR(45) -> Masukan dari user/sistem var_iteration:VARCHAR(45) -> Masukan dari user/sistem var_building_id:INT var_attribute_id:INT var_value_id:INT var_interlude_id:INT var_iteration_id:INT	
Deskripsi	
1	MULAI
2	
3	SET var_building_id = get_building_id(var_building);
4	
5	SET var_attribute_id =
6	get_attribute_id(var_attribute);
7	
8	SET var_value_id =
9	get_integer_variables_id(var_value);
10	
11	SET var_interlude_id =
12	get_datetime_variables_id(var_interlude);
13	
14	SET var_iteration_id =
15	get_varchar_variables_id(var_iteration);
16	
17	INSERT INTO resources_buildings
18	SET
19	building = var_building_id,
20	attribute = var attribute id,

21	value_iv = var_value_id,
22	interlude_dv = var_interlude_id,
23	iteration_vv = var_iteration_id;
24	
25	SELESAI

Penjelasan dari *procedure* insert_resource_building pada Tabel 4.34 yaitu:

1. Masukan dari *procedure* dalam tipe data VARCHAR.
2. Masukan berupa nama bangunan, nama jenis sumber daya, nama variabel INTEGER yang mewakili jumlah sumber daya yang dihasilkan, nama variabel DATETIME yang mewakili tempo penghasilan sumber daya dan nama variabel VARCHAR yang mewakili jenis perulangan dalam menghasilkan sumber daya.
3. Nama bangunan, nama jenis sumber daya dan nama-nama variabel nantinya akan dicari ID-nya untuk dimasukkan dalam tabel resources_buildings.

4.3.2.12 Implementasi Procedure insert_troop_build_time

Procedure yang digunakan untuk memasukkan data waktu lama pelatihan pasukan pertahanan atau penyerangan desa. Data nantinya akan dimasukkan dalam tabel troops_build_times.

Tabel 4.35 Pseudocode Procedure insert_troop_build_time

Pseudocode Procedure insert_troop_build_time	
Deklarasi awal	
var_troop:VARCHAR(50) -> Masukan dari user/sistem	
var_build_time:VARCHAR(50) -> Masukan dari user/sistem	
var_troop_id:INT	
var_datetime_id:INT	
Deskripsi	
1	MULAI
2	

3	SET var_troop_id = get_troop_id(var_troop);
4	
5	SET var_datetime_id =
6	get_datetime_variables_id(var_build_time);
7	
8	INSERT INTO troops_build_times
9	SET
10	troop = var_troop_id,
11	build_time_dv = var_datetime_id;
12	
13	SELESAI

Penjelasan dari *procedure* insert_troop_build_time pada tabel 4.35 yaitu:

1. Masukan dari *procedure* dalam tipe data VARCHAR.
2. Masukan berupa nama pasukan dan nama variabel DATETIME yang mewakili tempo lama pelatihan pasukan.
3. Nama pasukan dan nama variabel nantinya akan dicari ID-nya untuk dimasukkan dalam tabel troops_build_times.

4.3.2.13 Implementasi Procedure insert_troop_price

Procedure yang digunakan untuk memasukkan data harga tiap pasukan dalam permainan. Data nantinya akan dimasukkan dalam tabel troops_prices.

Tabel 4.36 Pseudocode Procedure insert_troop_price

Pseudocode Procedure insert_troop_price
Deklarasi awal var_troop_name:VARCHAR(50) -> Masukan dari user/sistem var_attribute:VARCHAR(50) -> Masukan dari user/sistem var_value_iv:VARCHAR(50) -> Masukan dari user/sistem var_troop_id:INT var_attribute_id:INT var_value_iv_id:INT Deskripsi

1	MULAI
2	
3	SET var_troop_id = get_troop_id(var_troop_name);
4	
5	SET var_attribute_id =
6	get_attribute_id(var_attribute);
7	
8	SET var_value_iv_id =
9	get_integer_variables_id(var_value_iv);
10	
11	INSERT INTO troops_prices
12	SET
13	troop = var_troop_id,
14	attribute = var_attribute_id,
15	value_iv = var_value_iv_id;
16	
17	SELESAI

Penjelasan dari *procedure* insert_troop_price pada tabel 4.36 yaitu:

1. Masukan dari *procedure* dalam tipe data VARCHAR.
2. Masukan berupa nama pasukan, nama jenis sumber daya dan nama variabel INTEGER yang mewakili harga pasukan.
3. Nama pasukan, nama jenis sumber daya dan nama variabel nantinya akan dicari ID-nya untuk dimasukkan dalam tabel troops_prices.

4.3.2.14 Implementasi Procedure insert_troop_stat

Procedure yang digunakan untuk memasukkan data atribut tiap pasukan dalam permainan. Data nantinya akan dimasukkan dalam tabel troops_stats.

Tabel 4.37 Pseudocode Procedure insert_troop_stat

Pseudocode Procedure insert_troop_stat
Deklarasi awal var_troop_name:VARCHAR(50) -> Masukan dari user/sistem var_attribute:VARCHAR(50) -> Masukan dari user/sistem var_value_iv:VARCHAR(50) -> Masukan dari user/sistem

<pre> var_troop_id:INT var_attribute_id:INT var_value_iv_id:INT Deskripsi </pre>	
1	MULAI
2	
3	SET var_troop_id = get_troop_id(var_troop_name);
4	
5	SET var_attribute_id =
6	get_attribute_id(var_attribute);
7	
8	SET var_value_iv_id =
9	get_integer_variables_id(var_value_iv);
10	
11	INSERT INTO troops_stats
12	SET
13	troop = var_troop_id,
14	attribute = var_attribute_id,
15	value_iv = var_value_iv_id;
16	
17	SELESAI

Penjelasan dari *procedure* insert_troop_stat pada tabel 4.37 yaitu:

1. Masukan dari *procedure* dalam tipe data VARCHAR.
2. Masukan berupa nama pasukan, nama jenis sumber daya dan nama variabel INTEGER yang mewakili atribut pasukan.
3. Nama pasukan, nama jenis sumber daya dan nama variabel nantinya akan dicari ID-nya untuk dimasukkan dalam tabel troops_stats.

4.3.2.15 Implementasi Procedure insert_village_location

Procedure yang digunakan untuk memasukkan data lokasi koordinat desa pemain dalam permainan. Data nantinya akan dimasukkan dalam tabel villages_locations.

Tabel 4.38 Pseudocode Procedure insert_village_location

Pseudocode Procedure insert_village_location	
Deklarasi awal	
var_idv INT -> Masukan dari user/sistem var_attribute VARCHAR(45) -> Masukan dari user/sistem var_value INT -> Masukan dari user/sistem var_ida INT	
Deskripsi	
1	MULAI
2	
3	SET var_ida := get_attribute_id(var_attribute);
4	
5	INSERT INTO
6	villages_locations(village,attribute,value)
7	VALUES(var_idv,var_ida,var_value);
8	
9	SELESAI

Penjelasan dari *procedure* insert_village_location pada tabel 4.38 yaitu:

1. Masukan dari *procedure* dalam tipe data VARCHAR untuk atribut dan INTEGER untuk ID desa dan koordinat posisi desa.
2. Nilai masukan untuk atribut ialah x_axis yang mewakili sumbu x atau y_axis yang mewakili sumbu y.
3. ID desa, nama atribut dan koordinat posisi desa akan dimasukkan dalam tabel villages_locations.

4.3.2.16 Implementasi Procedure insert_village_resource

Procedure yang digunakan untuk memberikan atau mengurangi sejumlah sumber daya dengan jenis tertentu pada desa pemain. Data nantinya akan dimasukkan dalam tabel villages_resources.

Tabel 4.39 Pseudocode Procedure insert_village_resource

Pseudocode Procedure insert_village_resource	
Deklarasi awal	
var_idv:INT -> Masukan dari user/sistem var_attribute:VARCHAR(45) -> Masukan dari user/sistem var_value:INT -> Masukan dari user/sistem var_execution_time:TIMESTAMP -> Masukan dari user/sistem var_ida:INT	
Deskripsi	
1	MULAI
2	
3	SET var_ida = get_attribute_id(var_attribute);
4	
5	INSERT INTO
6	villages_resources(village,attribute,value,execution_time)
7	VALUES(var_idv,var_ida,var_value,var_execution_time);
8	
9	SELESAI

Penjelasan dari *procedure* insert_village_resource pada tabel 4.39 yaitu:

1. Masukan dari *procedure* dalam tipe data VARCHAR untuk jenis sumber daya, TIMESTAMP untuk waktu eksekusi dan INTEGER untuk ID desa dan jumlah sumber daya yang akan diberikan.
2. nama jenis sumber daya nantinya akan dicari ID-nya untuk dimasukkan dalam tabel villages_resources bersama ID desa, jumlah sumber daya dan waktu eksekusi.

4.3.2.17 Implementasi Procedure insert_village_resource_schedule

Procedure yang digunakan untuk memasukkan data jadwal produksi sumber daya dari bangunan penghasil sumber daya dalam desa pemain. Data nantinya akan dimasukkan dalam tabel villages_resources_schedules.

Tabel 4.40 Pseudocode Procedure insert_village_resource_schedule

Pseudocode Procedure insert_village_resource_schedule	
Deklarasi awal	
var_idv:INT -> Masukan dari user/sistem var_attribute:VARCHAR(45) -> Masukan dari user/sistem var_value_iv:VARCHAR(45) -> Masukan dari user/sistem var_execution_time:DATETIME -> Masukan dari user/sistem var_interlude_dv:VARCHAR(45) -> Masukan dari user/sistem var_iteration_vv:VARCHAR(45) -> Masukan dari user/sistem var_attribute_id:INT var_value_id:INT var_interlude_id:INT var_iteration_id:INT	
Deskripsi	
1	MULAI
2	
3	SET var_attribute_id =
4	get_attribute_id(var_attribute);
5	
6	SET var_value_id =
7	get_integer_variables_id(var_value_iv);
8	
9	SET var_interlude_id = get_datetime_variables_id
10	(var_interlude_dv);
11	
12	SET var_iteration_id = get_varchar_variables_id
13	(var_iteration_vv);
14	
15	INSERT
16	INTO villages_resources_schedules(village,attribute,
17	value,execution_time,interlude,iteration)
18	VALUES (var_idv,var_attribute_id,var_value_id,
19	var_execution_time,var_interlude_id,var_iteration_id);
20	
21	SELESAI

Penjelasan dari *procedure* insert_village_resource_schedule pada tabel 4.40 yaitu:

1. Masukan dari *procedure* dalam tipe data INTEGER untuk ID desa, DATETIME untuk waktu eksekusi dan VARCHAR untuk nama jenis sumber daya serta nama-nama variabel.
2. Masukan berupa ID desa, nama jenis sumber daya, nama variabel INTEGER yang mewakili jumlah sumber daya yang dihasilkan, nama variabel DATETIME yang mewakili tempo penghasilan sumber daya dan nama variabel VARCHAR yang mewakili jenis perulangan dalam menghasilkan sumber daya.
3. Nama jenis sumber daya dan nama-nama variabel nantinya akan dicari ID-nya untuk dimasukkan dalam tabel `villages_resources_schedules`.

4.3.2.18 Implementasi Procedure `insert_village_troop`

Procedure yang digunakan untuk memberikan atau mengurangi sejumlah pasukan dengan jenis tertentu pada desa pemain dalam permainan. Data nantinya akan dimasukkan dalam tabel `villages_troops`.

Tabel 4.41 Pseudocode Procedure `insert_village_troop`

Pseudocode Procedure <code>insert_village_troop</code>	
Deklarasi awal <pre> var_village:INT -> Masukan dari user/sistem var_troop:VARCHAR(50) -> Masukan dari user/sistem var_value:INT -> Masukan dari user/sistem var_status:VARCHAR(50) -> Masukan dari user/sistem var_troop_id:INT var_status_id:INT var_status_away:INT = get_varchar_variables_id("troop_away") var_status_dead INT = get_varchar_variables_id("troop_dead") var_idvt INT = get_new_idvt() </pre> Deskripsi	
1	MULAI


```

2
3 SET var_troop_id = get_troop_id(var_troop);
4
5 SET var_status_id :=
6 get_varchar_variables_id(var_status);
7
8 IF (var_status_id = var_status_away AND var_value > 0)
9 OR (var_status_id = var_status_dead AND var_value > 0)
10 THEN
11     SET var_value = var_value * -1;
12 END IF;
13
14 INSERT INTO villages_troops
15 SET
16 idvt = var_idvt,
17 village = var_village,
18 troop = var_troop_id,
19 value = var_value,
20 execution_time = NOW(),
21 status_vv = var_status_id;
22
23 SELESAI

```

Penjelasan dari *procedure* insert_village_troop pada tabel 4.41 yaitu:

1. Masukan dari *procedure* dalam tipe data VARCHAR untuk nama dan status pasukan yang akan diberikan serta INTEGER untuk ID desa dan jumlah pasukan yang akan diberikan.
2. Penjumlahan atau pengurangan pasukan didasari dari status pasukan yang diberikan.
3. Untuk penambahan digunakan status troop_home, sedangkan untuk pengurangan digunakan status troop_away atau troop_dead.

4.3.2.19 Implementasi Procedure manual_processing_event_and_schedulling

Procedure yang digunakan untuk melakukan pemrosesan terjadwal secara *manual* tanpa menggunakan fitur *event* pada MySQL.

Tabel 4.42 Pseudocode Procedure manual_processing_event_and_schedulling

Pseudocode Procedure manual_processing_event_and_schedulling	
Deklarasi awal	
Deskripsi	
1	MULAI
2	
3	CALL sync_all_mem_variables();
4	
5	CALL villages_buildings_builds_schedulling();
6	
7	CALL villages_resources_schedulling();
8	
9	CALL villages_troops_attacks_schedulling();
10	
11	CALL villages_troops_builds_schedulling();
12	
13	CALL wars_loots_schedulling();
14	
15	SELESAI

Penjelasan dari *procedure* manual_processing_event_and_schedulling pada tabel 4.42 yaitu:

1. Pada prinsipnya, cara kerja *procedure* ini ialah memanggil *procedure-procedure* penjadwalan yang telah dibuat.
2. *Procedure-procedure* yang dipanggil sudah dipanggil secara otomatis melalui fitur *event*.

4.3.2.20 Implementasi Procedure order_attack

Procedure yang digunakan pemain untuk mengirim pasukan yang ada di desa pemain guna menyerang serta merampas sumber daya dari desa lain.

Tabel 4.43 Pseudocode Procedure order_attack

Pseudocode Procedure order_attack

Deklarasi awal

```
var_village:INT -> Masukan dari user/sistem
var_troop:VARCHAR(50) -> Masukan dari user/sistem
var_target_village:INT -> Masukan dari user/sistem
var_process:VARCHAR(50)
var_next_process:VARCHAR(50)
var_marker_a:INT
var_troop_type:VARCHAR(50)
var_troop_id:INT
var_troop_amount:INT
var_marker_k:INT
var_attack:INT
var_idvt:INT
var_status_away:INT
var_verificator:INT
var_speedless_troop:INT
var_speed_value:INT
var_attribute_id:INT
var_travel_time:TIME
var_status_id:INT
```

Deskripsi

1	MULAI
2	
3	SET var_attack = get_new_attack();
4	
5	REPEAT
6	
7	SET var_marker_a = LOCATE("&",var_troop);
8	
9	IF var_marker_a != 0 THEN
10	SET var_process = SUBSTRING(var_troop,1,
11	var_marker_a - 1);
12	SET var_next_process = SUBSTRING(var_troop,
13	var_marker_a + 1);
14	ELSE
15	SET var_process = SUBSTRING(var_troop,
16	var_marker_a + 1);
17	END IF;
18	
19	SET var_troop = var_next_process;
20	
21	SET var_marker_k = LOCATE("#",var_process);
22	


```

23 SET var_troop_type = SUBSTRING(var_process,1,
24 var_marker_k - 1);
25
26 SET var_troop_amount = SUBSTRING(var_process,
27 var_marker_k + 1);
28
29 SET var_troop_id = get_troop_id(var_troop_type);
30
31 SELECT IFNULL(SUM(value),0) INTO
32 var_verificator FROM villages_troops
33 WHERE village = var_village AND troop = var_troop_id;
34
35 IF var_troop_amount > var_verificator THEN
36     SET SQL_SAFE_UPDATES=0;
37
38     DELETE villages_troops,villages_troops_attacks
39     FROM villages_troops
40     INNER JOIN villages_troops_attacks
41     ON villages_troops.idvt = villages_troops_
42     attacks.village_troop
43     WHERE villages_troops_attacks.attack =
44     var_attack;
45
46     SET SQL_SAFE_UPDATES=1;
47
48     SET var_marker_a = 0;
49 ELSE
50     IF var_troop_amount != 0 THEN
51         SET var_idvt = get_new_idvt();
52
53         SET var_status_away =
54         get_varchar_variables_id("troop_away");
55
56         INSERT INTO villages_troops
57         SET
58         idvt = var_idvt,
59         village = var_village,
60         troop = var_troop_id,
61         value = var_troop_amount * -1,
62         execution_time = NOW(),
63         status_vv = var_status_away;
64
65         INSERT INTO villages_troops_attacks
66         SET
67         attack = var_attack,
68         village_troop = var_idvt;
69     END IF;

```

```

70      END IF;
71
72      UNTIL var_marker_a = 0 END REPEAT;
73
74      SET var_attribute_id =
75      get_attribute_id("troop_speed");
76
77      SELECT vt.troop,MIN(iv.value) INTO
78      var_speedless_troop,
79      var_speed_value
80      FROM villages_troops_attacks vta
81      LEFT JOIN villages_troops vt ON vt.idvt =
82      vta.village_troop
83      LEFT JOIN troops_stats ts ON ts.troop = vt.troop
84      LEFT JOIN mem_integer_variables iv ON iv.idiv =
85      ts.value_iv
86      WHERE ts.attribute = var_attribute_id
87      AND vta.attack = var_attack;
88
89      IF var_speedless_troop IS NOT NULL THEN
90          SET var_travel_time =
91          get_travel_time(var_speedless_troop,var_village,
92          var_target_village);
93
94          SET var_status_id =
95          get_varchar_variables_id("troop_away");
96
97          INSERT INTO villages_troops_attacks_schedules
98          SET
99          village_troop_attack = var_attack,
100         target_village = var_target_village,
101         execution_time = ADDTIME(NOW(),var_travel_time),
102         status_vv = var_status_id;
103      END IF;
104
105      SELESAI

```

Penjelasan dari *procedure* order_attack pada tabel 4.43 yaitu:

1. Dikarenakan tidak adanya variabel *array* pada MySQL maka perlu dibuat mekanisme pengganti yang mampu menggantikan fungsi variabel *array* yang diperlukan dalam perangkat lunak.

2. Mekanisme pengganti yang dipilih ialah penggunaan variabel VARCHAR sebagai penampung kumpulan nilai yang dibutuhkan.
3. Nantinya kumpulan nilai dalam variabel VARCHAR akan diuraikan berdasarkan karakter tertentu yakni karakter '#' dan karakter '&'.
4. Format pengkodean kumpulan nilai ialah :
`"jenis_pasukan#jumlah_pasukan&jenis_pasukan#jumlah_pasukan&..."`.
5. Penguraian variabel pengganti menggunakan fitur perulangan, *function* LOCATE dan *function* SUBSTRING dalam MySQL seperti pada baris 5-27.
6. Seleksi kondisi pada baris 35-70 bertujuan untuk melakukan verifikasi jumlah pasukan yang akan dikirim apakah lebih besar dari jumlah pasukan yang ada di desa.
7. Manakala pasukan yang akan dikirim berjumlah lebih besar, maka serangan akan dibatalkan karena gagal dalam proses verifikasi.
8. Pada baris 74-103 akan dilakukan proses *generate* waktu tempuh.
9. Waktu tempuh didapat dari jarak dibagi kecepatan pasukan terlambat yang ikut dalam penyerangan.

4.3.2.21 Implementasi Procedure order_attack_simulation

Procedure yang digunakan pengembang untuk mengirim pasukan yang ada di desa pemain guna menyerang serta merampas sumber daya dari desa lain.

Tabel 4.44 Pseudocode Procedure order_attack_simulation

Pseudocode Procedure order_attack_simulation
Deklarasi awal <pre> var_village:INT -> Masukan dari user/sistem var_troop:VARCHAR(50) -> Masukan dari user/sistem var_target_village:INT -> Masukan dari user/sistem var_process:VARCHAR(50) var_next_process:VARCHAR(50) var marker a:INT </pre>


```

var_troop_type:VARCHAR(50)
var_troop_id:INT
var_troop_amount:INT
var_marker_k:INT
var_attack:INT
var_idvt:INT
var_status_away:INT
var_verificator:INT
var_speedless_troop:INT
var_speed_value:INT
var_attribute_id:INT
var_travel_time:TIME
var_status_id:INT

```

Deskripsi

1	MULAI
2	
3	SET var_attack = get_new_attack();
4	
5	REPEAT
6	
7	SET var_marker_a = LOCATE("&",var_troop);
8	
9	IF var_marker_a != 0 THEN
10	SET var_process = SUBSTRING(var_troop,1,
11	var_marker_a - 1);
12	SET var_next_process = SUBSTRING(var_troop,
13	var_marker_a + 1);
14	ELSE
15	SET var_process = SUBSTRING(var_troop,
16	var_marker_a + 1);
17	END IF;
18	
19	SET var_troop = var_next_process;
20	
21	SET var_marker_k = LOCATE("#",var_process);
22	
23	SET var_troop_type = SUBSTRING(var_process,1,
24	var_marker_k - 1);
25	
26	SET var_troop_amount = SUBSTRING(var_process,
27	var_marker_k + 1);
28	
29	SET var_troop_id = get_troop_id(var_troop_type);
30	
31	SELECT IFNULL(SUM(value),0) INTO

```
32 var_verificator FROM villages_troops
33 WHERE village = var_village AND troop = var_troop_id;
34
35 IF var_troop_amount > var_verificator THEN
36     SET SQL_SAFE_UPDATES=0;
37
38     DELETE villages_troops,villages_troops_attacks
39     FROM villages_troops
40     INNER JOIN villages_troops_attacks
41     ON villages_troops.idvt = villages_troops_
42     attacks.village_troop
43     WHERE villages_troops_attacks.attack =
44     var_attack;
45
46     SET SQL_SAFE_UPDATES=1;
47
48     SET var_marker_a = 0;
49 ELSE
50     IF var_troop_amount != 0 THEN
51         SET var_idvt = get_new_idvt();
52
53         SET var_status_away =
54         get_varchar_variables_id("troop_away");
55
56         INSERT INTO villages_troops
57         SET
58         idvt = var_idvt,
59         village = var_village,
60         troop = var_troop_id,
61         value = var_troop_amount * -1,
62         execution_time = NOW(),
63         status_vv = var_status_away;
64
65         INSERT INTO villages_troops_attacks
66         SET
67         attack = var_attack,
68         village_troop = var_idvt;
69     END IF;
70 END IF;
71
72 UNTIL var_marker_a = 0 END REPEAT;
73
74 SET var_attribute_id =
75 get_attribute_id("troop_speed");
76
77 SELECT vt.troop,MIN(iv.value) INTO
78 var speedless troop,
```

```

79  var_speed_value
80  FROM villages_troops_attacks vta
81  LEFT JOIN villages_troops vt ON vt.idvt =
82  vta.village_troop
83  LEFT JOIN troops_stats ts ON ts.troop = vt.troop
84  LEFT JOIN mem_integer_variables iv ON iv.idiv =
85  ts.value_iv
86  WHERE ts.attribute = var_attribute_id
87  AND vta.attack = var_attack;
88
89  IF var_speedless_troop IS NOT NULL THEN
90      SET var_travel_time =
91      get_travel_time(var_speedless_troop,var_village,
92      var_target_village);
93
94      SET var_status_id =
95      get_varchar_variables_id("troop_away");
96
97      INSERT INTO villages_troops_attacks_schedules
98      SET
99      village_troop_attack = var_attack,
100     target_village = var_target_village,
101     execution_time = NOW(),
102     status_vv = var_status_id;
103  END IF;
104
105  SELESAI

```

Penjelasan dari *procedure* order_attack pada tabel 4.44 yaitu:

1. Merupakan *procedure* yang dipakai pengembang untuk melakukan pengujian kinerja *procedure* order_attack.
2. Prinsip kerja *procedure* kurang lebih sama dengan *procedure* order_attack, namun yang menjadi pembeda ialah proses memasukkan data execution_time pada baris 101 yang nilainya diambil dari *function* NOW.
3. Fungsi *procedure* ialah agar pengembang tidak memerlukan waktu lama dalam menguji *procedure* villages_troops_attacks_scheduling.

4.3.2.2 Implementasi Procedure order_building

Procedure yang digunakan pemain untuk mendirikan bangunan, baik bangunan penghasil sumber daya maupun bangunan non-penghasil sumber daya pada desa pemain.

Tabel 4.45 Pseudocode Procedure order_building

Pseudocode Procedure order_building	
Deklarasi awal <pre> var_village:INT -> Masukan dari user/sistem var_building:VARCHAR(50) -> Masukan dari user/sistem var_building_id:INT var_verification:INT var_verificator:INT var_building_build_slot:INT var_slot:INT var_used_slot:INT var_status_vv_id:INT </pre> Deskripsi	
1	MULAI
2	
3	SET var_building_build_slot =
4	get_integer_variables("building_build_slot");
5	
6	SET var_building_id = get_building_id("town_center");
7	
8	SELECT COUNT(building) * var_building_build_slot INTO
9	var_slot
10	FROM villages_buildings
11	WHERE village = var_village AND building =
12	var_building_id;
13	
14	SET var_status_vv_id =
15	get_varchar_variables_id("iteration_once");
16	
17	SELECT COUNT(building) + 1 INTO var_used_slot
18	FROM villages_buildings_builds_schedules
19	WHERE village = var_village AND status_vv =
20	var_status_vv_id;
21	

```

22 IF var_used_slot <= var_slot THEN
23     SET var_building_id =
24     get_building_id(var_building);
25
26     SELECT COUNT(idbp) INTO var_verification
27     FROM buildings_prices
28     WHERE building = var_building_id;
29
30     SELECT COUNT(bp.building) INTO var_verificator
31     FROM buildings_prices bp
32     LEFT JOIN mem_integer_variables iv ON iv.idiv =
33     bp.value_iv
34     LEFT JOIN
35     (
36         SELECT village,attribute,SUM(value) AS
37         resource
38         FROM villages_resources
39         GROUP BY village,attribute
40         HAVING village = var_village
41     ) AS va ON va.attribute = bp.attribute
42     WHERE bp.building = var_building_id AND
43     va.resource >= iv.value;
44
45     IF var_verification = var_verificator THEN
46     CALL buy_building(var_village,var_building_id);
47     END IF;
48
49 END IF;
50
51 SELESAI

```

Penjelasan dari *procedure* order_building pada tabel 4.45 yaitu:

1. Seleksi kondisi pada baris 22 berfungsi untuk melakukan verifikasi adanya jumlah slot pembangunan yang masih tersedia.
2. Pembangunan bangunan secara simultan dibatasi dalam jumlah tertentu untuk setiap desa.
3. Jumlah slot pembangunan bangunan bergantung dari jumlah town_center dan jumlah slot pembangunan setiap town_center.
4. Seleksi kondisi pada baris 45 berfungsi untuk melakukan verifikasi ketersediaan sumber daya untuk melakukan pembangunan.

4.3.2.23 Implementasi Procedure order_troop

Procedure yang digunakan pemain untuk melakukan pelatihan pasukan pertahanan maupun pasukan penyerang untuk desa pemain.

Tabel 4.46 Pseudocode Procedure order_troop

Pseudocode Procedure order_troop	
Deklarasi awal	
var_village:INT -> Masukan dari user/sistem var_troop:VARCHAR(50) -> Masukan dari user/sistem var_troop_amount:INT -> Masukan dari user/sistem var_troop_id:INT var_building_id:INT var_verification:INT var_verificator:INT var_troop_build_slot:INT var_slot:INT var_used_slot:INT var_status_vv_id:INT	
Deskripsi	
1	MULAI
2	
3	SET var_troop_build_slot =
4	get_integer_variables("troop_build_slot");
5	
6	SET var_building_id = get_building_id("barrack");
7	
8	SELECT COUNT(building) * var_troop_build_slot INTO
9	var_slot
10	FROM villages_buildings
11	WHERE village = var_village AND building =
12	var_building_id;
13	
14	SET var_status_vv_id =
15	get_varchar_variables_id("iteration_once");
16	
17	SELECT IFNULL(SUM(value),0) + var_troop_amount INTO
18	var used slot


```

19 FROM villages_troops_builds_schedules
20 WHERE village = var_village AND status_vv =
21 var_status_vv_id;
22
23 IF var_used_slot <= var_slot THEN
24     SET var_troop_id := get_troop_id(var_troop);
25
26     SELECT COUNT(idtp) INTO var_verification
27     FROM troops_prices
28     WHERE troop = var_troop_id;
29
30     SELECT COUNT(tp.troop) INTO var_verificator
31     FROM troops_prices tp
32     LEFT JOIN mem_integer_variables iv ON iv.idiv =
33     tp.value_iv
34     LEFT JOIN
35     (
36         SELECT village,attribute,SUM(value) AS
37         resource
38         FROM villages_resources
39         GROUP BY village,attribute
40         HAVING village = var_village
41     ) AS va ON va.attribute = tp.attribute
42     WHERE tp.troop = var_troop_id AND va.resource >=
43     iv.value * var_troop_amount;
44
45     IF var_verification = var_verificator THEN
46     CALL
47     buy_troop(var_village,var_troop_id,var_troop_amount);
48     END IF;
49
50 END IF;
51
52 SELESAI

```

Penjelasan dari *procedure* order_troop pada tabel 4.46 yaitu:

1. Seleksi kondisi pada baris 23 berfungsi untuk melakukan verifikasi adanya jumlah slot pelatihan pasukan yang masih tersedia.
2. Pelatihan pasukan secara simultan dibatasi dalam jumlah tertentu untuk setiap desa.
3. Jumlah slot pelatihan pasukan bergantung dari jumlah barrack dan jumlah slot pembangunan setiap barrack.

4. Seleksi kondisi pada baris 45 berfungsi untuk melakukan verifikasi ketersediaan sumber daya untuk melakukan pelatihan.

4.3.2.24 Implementasi Procedure player_registration

Procedure yang digunakan calon pemain untuk melakukan pendaftaran pada *game server online* sebelum memulai permainan.

Tabel 4.47 Pseudocode Procedure player_registration

Pseudocode Procedure player_registration	
Deklarasi awal	
var_username:varchar(45) -> Masukan dari user/sistem	
var_password:varchar(45) -> Masukan dari user/sistem	
Deskripsi	
1	MULAI
2	
3	INSERT INTO players
4	SET
5	username = var_username,
6	password = var_password;
7	
8	SELESAI

Penjelasan dari *procedure* player_registration pada tabel 4.47 yaitu:

1. Masukan username pemain tidak boleh sama dengan pemain yang sudah lebih dahulu terdaftar pada *game server online*.
2. Untuk algoritma jumlah karakter minimal pada username dan password diserahkan sepenuhnya pada *game client*.

4.3.2.25 Implementasi Procedure random_village_axis

Salah satu bagian *procedure* dari proses pendaftaran pemain yang dipakai untuk menetapkan koordinat lokasi desa pemain. Pemakaian hanya berlangsung sekali saat pemain melakukan pendaftaran.

Tabel 4.48 Pseudocode Procedure random_village_axis

Pseudocode Procedure random_village_axis	
Deklarasi awal <pre> var_x_axis INT -> Keluaran dari sistem var_y_axis INT -> Keluaran dari sistem var_map_size_x_min:INT var_map_size_x_max:INT var_map_size_y_min:INT var_map_size_y_max:INT var_verification:INT var_attribute_x_axis:INT var_attribute_y_axis:INT </pre> Deskripsi	
1	MULAI
2	
3	SET var_map_size_x_min =
4	get_integer_variables("map_size_x_min");
5	
6	SET var_map_size_x_max =
7	get_integer_variables("map_size_x_max");
8	
9	SET var_map_size_y_min =
10	get_integer_variables("map_size_y_min");
11	
12	SET var_map_size_y_max =
13	get_integer_variables("map_size_y_max");
14	
15	SET var_attribute_x_axis = get_attribute_id("x_axis");
16	
17	SET var_attribute_y_axis = get_attribute_id("y_axis");
18	
19	REPEAT
20	
21	SET var x axis =


```

22 random_integer(var_map_size_x_min,var_map_size_x_max);
23
24 SET var_y_axis =
25 random_integer(var_map_size_y_min,var_map_size_y_max);
26
27 SELECT COUNT(*) INTO var_verification
28 FROM
29 (
30     SELECT village, COUNT(village) as summary
31     FROM villages_locations
32     WHERE (attribute = var_attribute_x_axis AND value
33           = var_x_axis)
34     OR (attribute = var_attribute_y_axis AND value =
35         var_y_axis)
36     GROUP BY village
37 )
38 AS cek WHERE summary > 1;
39
40 UNTIL var_verification = 0
41
42 END REPEAT;
43
44 SELESAI

```

Penjelasan dari *procedure* random_village_axis pada tabel 4.48 yaitu:

1. Batas atas dan batas bawah baik sumbu x maupun sumbu y mewakili luas peta permainan yang panjang dan lebarnya sudah ditentukan sebelumnya.
2. Prinsip kerja *procedure* ialah memanggil *function* random_integer berulang kali untuk mendapatkan koodinat posisi calon pemain baik koordinat sumbu x maupun koordinat sumbu y.
3. Perulangan akan berhenti ketika hasil *function* random_integer terhadap lokasi desa calon pemain berbeda dengan lokasi lokasi desa pemain yang sudah terdaftar terlebih dahulu.
4. Verifikasi koordinat desa calon pemain dilakukan pada baris 27-38.

4.3.2.26 Implementasi Procedure reset_game_server

Procedure yang digunakan *administrator* atau pengembang untuk melakukan *reset* permainan pada *game server online*.

Tabel 4.49 Pseudocode Procedure reset_game_server

Pseudocode Procedure reset_game_server	
Deklarasi awal	
Deskripsi	
1	MULAI
2	
3	TRUNCATE war_loots_schedules;
4	
5	TRUNCATE villages_troops_attacks_schedules;
6	
7	TRUNCATE villages_troops_attacks;
8	
9	TRUNCATE villages_troops_builds_schedules;
10	
11	TRUNCATE villages_troops;
12	
13	TRUNCATE villages_locations;
14	
15	TRUNCATE villages_buildings_builds_schedules;
16	
17	TRUNCATE villages_buildings_resources_schedules;
18	
19	TRUNCATE villages_resources_schedules;
20	
21	TRUNCATE villages_buildings;
22	
23	TRUNCATE villages_resources;
24	
25	TRUNCATE villages;
26	
27	TRUNCATE players;
28	
29	SELESAI

Penjelasan dari *procedure* reset_game_server pada tabel 4.49 yaitu:

1. Prinsip kerja ialah mengosongkan beberapa tabel dalam database *game server online*.
2. Tabel yang akan dikosongkan ialah tabel-tabel yang berisi data pemain.
3. Pengosongan tabel menggunakan fitur TRUNCATE MySQL.

4.3.2.27 Implementasi Procedure sync_all_mem_variables

Procedure yang digunakan *administrator*, pengembang atau sistem untuk melakukan sinkronisasi antara seluruh tabel variabel dengan tabel variabel *memory engine*.

Tabel 4.50 Pseudocode Procedure sync_all_mem_variables

Pseudocode Procedure sync_all_mem_variables	
Deklarasi awal	
Deskripsi	
1	MULAI
2	
3	CALL sync_integer_variables;
4	
5	CALL sync_datetime_variables;
6	
7	CALL sync_varchar_variables;
8	
9	SELESAI

Penjelasan dari *procedure* sync_all_mem_variabel pada tabel 4.50 yaitu:

1. Prinsip kerja ialah pemanggilan seluruh *procedure* sinkronisasi tabel variabel yang terdapat pada *game server online*.
2. *Procedure* akan dipanggil secara rutin dan terjadwal melalui fitur *event*.

4.3.2.28 Implementasi Procedure sync_datetime_variables

Procedure yang digunakan *administrator*, pengembang atau sistem untuk melakukan sinkronisasi antara tabel variabel datetime dengan tabel variabel datetime *memory engine*.

Tabel 4.51 Pseudocode Procedure sync_datetime_variables

Pseudocode Procedure sync_datetime_variables	
Deklarasi awal <pre> var_iddv:INT var_variable:VARCHAR(45) var_value:DATETIME var_finished:INT = 0 var_verification:INT curs:CURSOR FOR SELECT iddv,variable,value FROM datetime_variables; DECLARE CONTINUE HANDLER FOR NOT FOUND SET var_finished = 1 </pre> Deskripsi	
1	MULAI
2	
3	OPEN curs;
4	
5	sync_process: LOOP
6	
7	FETCH curs INTO var_iddv,var_variable,var_value;
8	
9	IF var_finished = 1 THEN
10	LEAVE sync_process;
11	END IF;
12	
13	SELECT iddv INTO var_verification
14	FROM mem_datetime_variables
15	WHERE iddv = var_iddv;
16	
17	IF var_verification IS NULL THEN
18	INSERT INTO
19	mem_datetime_variables(iddv,variable,value)
20	VALUES (var_iddv,var_variable,var_value);
21	

```

22      SET var_finished = 0;
23  END IF;
24
25  SET var_verification := NULL;
26
27  END LOOP sync_process;
28
29  CLOSE curs;
30
31  SELESAI
    
```

Penjelasan dari *procedure* *sync_datetime_variabel* pada tabel 4.51 yaitu:

1. Prinsip kerja ialah pemasukan data yang terdapat pada tabel variabel DATETIME ke dalam tabel dengan *memory engine* manakala data tersebut tidak ditemukan dalam tabel dengan *memory engine*.
2. Deklarasi awal curs digunakan untuk mendapatkan keseluruhan data pada tabel yang kemudian pada perulangan di baris 5-27 dilakukan proses verifikasi serta pemasukan data tatkala tidak ditemukannya suatu data pada tabel dengan *memory engine*.
3. *Procedure* akan dipanggil pada *procedure* *sync_all_mem_variables*.

4.3.2.29 Implementasi Procedure *sync_integer_variables*

Procedure yang digunakan *administrator*, pengembang atau sistem untuk melakukan sinkronisasi antara tabel variabel integer dengan tabel variabel integer *memory engine*.

Tabel 4.52 Pseudocode Procedure *sync_integer_variables*

Pseudocode Procedure <i>sync_integer_variables</i>
Deklarasi awal var_idiv:INT var_variable:VARCHAR(45) var_value:INT var_finished:INT = 0

```
var_verification:INT
curs:CURSOR FOR
    SELECT idiv,variable,value FROM
        integer_variables;
DECLARE CONTINUE HANDLER FOR NOT FOUND SET var_finished = 1
```

Deskripsi

```
1  MULAI
2
3  OPEN curs;
4
5  sync_process: LOOP
6
7  FETCH curs INTO var_idiv,var_variable,var_value;
8
9  IF var_finished = 1 THEN
10     LEAVE sync_process;
11 END IF;
12
13 SELECT idiv INTO var_verification
14 FROM mem_integer_variables
15 WHERE idiv = var_idiv;
16
17 IF var_verification IS NULL THEN
18     INSERT INTO
19     mem_integer_variables(idiv,variable,value)
20     VALUES (var_idiv,var_variable,var_value);
21
22     SET var_finished = 0;
23 END IF;
24
25 SET var_verification := NULL;
26
27 END LOOP sync_process;
28
29 CLOSE curs;
30
31 SELESAI
```

Penjelasan dari *procedure* sync_integer_variabel pada tabel 4.52 yaitu:

1. Prinsip kerja ialah pemasukan data yang terdapat pada tabel variabel INTEGER ke dalam tabel dengan *memory engine* manakala data tersebut tidak ditemukan dalam tabel dengan *memory engine*.
2. Deklarasi awal curs digunakan untuk mendapatkan keseluruhan data pada tabel yang kemudian pada perulangan di baris 5-27 dilakukan proses verifikasi serta pemasukan data tatkala tidak ditemukannya suatu data pada tabel dengan *memory engine*.
3. *Procedure* akan dipanggil pada *procedure sync_all_mem_variables*.

4.3.2.30 Implementasi Procedure sync_varchar_variables

Procedure yang digunakan *administrator*, pengembang atau sistem untuk melakukan sinkronisasi antara tabel variabel varchar dengan tabel variabel varchar *memory engine*.

Tabel 4.53 Pseudocode Procedure sync_varchar_variables

Pseudocode Procedure sync_varchar_variables	
Deklarasi awal <pre> var_idvv:INT var_variable:VARCHAR(45) var_value: VARCHAR(45) var_finished:INT = 0 var_verification:INT curs:CURSOR FOR SELECT idvv,variable,value FROM varchar_variables; DECLARE CONTINUE HANDLER FOR NOT FOUND SET var_finished = 1 </pre> Deskripsi	
1	MULAI
2	
3	OPEN curs;
4	
5	sync_process: LOOP
6	

```
7  FETCH curs INTO var_idvv,var_variable,var_value;
8
9  IF var_finished = 1 THEN
10     LEAVE sync_process;
11 END IF;
12
13 SELECT idvv INTO var_verification
14 FROM mem_varchar_variables
15 WHERE idvv = var_idvv;
16
17 IF var_verification IS NULL THEN
18     INSERT INTO
19     mem_varchar_variables(idvv,variable,value)
20     VALUES (var_idvv,var_variable,var_value);
21
22     SET var_finished = 0;
23 END IF;
24
25 SET var_verification := NULL;
26
27 END LOOP sync_process;
28
29 CLOSE curs;
30
31 SELESAI
```

Penjelasan dari *procedure* sync_varchar_variabel pada tabel 4.53 yaitu:

1. Prinsip kerja ialah pemasukan data yang terdapat pada tabel variabel VARCHAR ke dalam tabel dengan *memory engine* manakala data tersebut tidak ditemukan dalam tabel dengan *memory engine*.
2. Deklarasi awal curs digunakan untuk mendapatkan keseluruhan data pada tabel yang kemudian pada perulangan di baris 5-27 dilakukan proses verifikasi serta pemasukan data tatkala tidak ditemukannya suatu data pada tabel dengan *memory engine*.
3. *Procedure* akan dipanggil pada *procedure* sync_all_mem_variables.

4.3.2.31 Implementasi Procedure village_building_registration

Procedure yang digunakan sistem untuk memasukkan data pada tabel `villages_buildings` tatkala dilakukan pembangunan bangunan.

Tabel 4.54 Pseudocode Procedure village_building_registration

Pseudocode Procedure village_building_registration	
Deklarasi awal	
var_village:INT -> Masukan dari user/sistem var_building:INT -> Masukan dari user/sistem var_amount:INT -> Masukan dari user/sistem var_iteration INT = 0	
Deskripsi	
1	MULAI
2	
3	WHILE var_iteration < var_amount AND var_amount > 0 DO
4	INSERT INTO villages_buildings
5	SET
6	village = var_village,
7	building = var_building;
8	
9	SET var_iteration = var_iteration + 1;
10	END WHILE;
11	
12	SELESAI

Penjelasan dari *procedure* village_building_registration pada tabel 4.54 yaitu:

1. Mekanisme pemasukan data ke dalam tabel `villages_buildings` harus melalui *procedure* ini.
2. Perulangan pada baris 3 berfungsi untuk mengulang proses pemasukan data yang digunakan untuk membangun bangunan dengan jumlah sesuai dengan masukan.

4.3.2.32 Implementasi Procedure village_registration

Salah satu bagian *procedure* dari proses pendaftaran pemain yang dipakai untuk mendaftarkan desa pemain pada tabel *villages* serta mengisi desa pemain dengan bangunan dan sumber daya *default*.

Tabel 4.55 Pseudocode Procedure village_registration

Pseudocode Procedure village_registration	
Deklarasi awal	
var_player:INT -> Masukan dari user/sistem var_village_name:VARCHAR(45) -> Masukan dari user/sistem var_x_axis:INT -> Masukan dari user/sistem var_y_axis:INT -> Masukan dari user/sistem var_idv:INT	
Deskripsi	
1	MULAI
2	
3	SET var_idv = get_new_idv();
4	
5	INSERT INTO villages(idv,player,village_name)
6	VALUES(var_idv,var_player,var_village_name);
7	
8	CALL default_village_resources(var_idv,var_x_axis,
9	var_y_axis,NOW());
10	
11	CALL default_village_buildings(var_idv);
12	
13	SELESAI

Penjelasan dari *procedure* village_registration pada tabel 4.55 yaitu:

1. *Procedure* ditempatkan dalam *trigger* pada tabel *temp_villages* yang menggunakan *blackhole engine*.
2. Nilai-nilai masukan diambil dari data-data yang dimasukkan pada tabel *temp_villages*.

3. Procedure akan digunakan dalam *trigger* pada tabel BLACKHOLE temp_villages.

4.3.2.33 Implementasi Procedure villages_buildings_builds_scheduling

Procedure yang digunakan *administrator*, pengembang atau sistem untuk memroses data jadwal pembangunan bangunan yang ada pada tabel villages_buildings_builds_schedules.

Tabel 4.56 Pseudocode Procedure villages_buildings_builds_scheduling

Pseudocode Procedure villages_buildings_builds_scheduling	
Deklarasi awal var_idvbbs:INT var_village:INT var_building:INT var_finished_build:TIMESTAMP var_status_req:INT = get_varchar_variables_id ("iteration_done"); var_finished:INT = 0 var_status_done:INT curs:CURSOR FOR SELECT idvbbs,village,building,finished,status_vv FROM villages_buildings_builds_schedules WHERE finished <= NOW() AND status_vv != var_status _req FOR UPDATE CONTINUE HANDLER FOR NOT FOUND SET var_finished = 1	
Deskripsi	
1	MULAI
2	
3	OPEN curs;
4	
5	sync_process: LOOP
6	
7	FETCH curs INTO
8	var_idvbbs,var_village,var_building,var_finished_build,
9	var_status_vv;
10	
11	IF var_finished = 1 THEN

```
12      LEAVE sync_process;
13 END IF;
14
15 CALL
16 village_building_registration(var_village,var_building,1);
17
18 SET var_status_done :=
19 get_varchar_variables_id("iteration_done");
20
21 UPDATE villages_buildings_builds_schedules
22 SET status_vv := var_status_done
23 WHERE idvbbs = var_idvbbs;
24
25 END LOOP sync_process;
26
27 CLOSE curs;
28
29 SELESAI
```

Penjelasan dari *procedure* villages_buildings_builds_scheduling pada tabel 4.56 yaitu:

1. *Procedure* akan dipanggil secara rutin dan terjadwal melalui fitur *event*.
2. Data jadwal pembangunan yang diproses pada *procedure* ini ialah data yang memenuhi syarat yang telah ditetapkan pada algoritma SELECT dalam variabel curs di deklarasi awal.
3. Syarat yang harus dipenuhi ialah data dengan status tidak sama dengan "iteration_done" dan waktu penyelesaian pembangunan harus lebih kecil sama dengan waktu pemrosesan.
4. Algoritma SELECT pada variabel curs menggunakan fitur FOR UPDATE yang berfungsi untuk menutup akses SELECT yang lain terhadap data yang telah atau sedang diproses oleh SELECT sebelumnya.
5. Deklarasi awal curs digunakan untuk mendapatkan keseluruhan data yang memenuhi syarat pada tabel, kemudian pada perulangan di baris 5-25 dilakukan proses penyelesaian pembangunan bangunan pada desa pemain.

4.3.2.34 Implementasi Procedure villages_resources_scheduling

Procedure yang digunakan *administrator*, pengembang atau sistem untuk memproses data jadwal penghasilan sumber daya yang ada pada tabel *villages_resources_schedules*.

Tabel 4.57 Pseudocode Procedure villages_resources_scheduling

Pseudocode Procedure villages_resources_scheduling	
Deklarasi awal <pre> var_idvas:INT var_village:INT var_attribute:VARCHAR(45) var_value_iv:INT var_execution_time:TIMESTAMP var_interlude_dv:INT var_iteration_vv:INT var_iteration_req:INT = get_varchar_variables_id ("iteration_done") var_value:INT var_new_execution_time:DATETIME var_interlude:DATETIME var_finished INT = 0 var_check_iteration:INT var_iteration_done:INT curs:CORSOR FOR SELECT idvas,village,attribute,value_iv,execution_time, interlude_dv,iteration_vv FROM villages_resources_schedules WHERE execution_time <= NOW() AND iteration_vv != var_iteration_req FOR UPDATE CONTINUE HANDLER FOR NOT FOUND SET var_finished = 1 </pre> Deskripsi	
1	MULAI
2	
3	OPEN curs;
4	
5	sync_process: LOOP
6	
7	FETCH curs INTO

```
8  var_idvas,var_village,var_attribute,var_value_iv,  
9  var_execution_time,var_interlude_dv,var_iteration_vv;  
10  
11  IF var_finished = 1 THEN  
12      LEAVE sync_process;  
13  END IF;  
14  
15  SET var_value =  
16  get_integer_variables_from_id(var_value_iv);  
17  
18  INSERT INTO villages_resources  
19  SET  
20  village = var_village,  
21  attribute = var_attribute,  
22  value = var_value,  
23  execution_time = NOW();  
24  
25  SET var_check_iteration =  
26  get_varchar_variables_id("iteration_once");  
27  
28  IF var_iteration_vv = var_check_iteration THEN  
29      SET var_iteration_done =  
30      get_varchar_variables_id("iteration_done");  
31  
32      UPDATE villages_resources_schedules  
33      SET iteration_vv = var_iteration_done  
34      WHERE idvas = var_idvas;  
35  END IF;  
36  
37  SET var_check_iteration =  
38  get_varchar_variables_id("iteration_frequently");  
39  
40  IF var_iteration_vv = var_check_iteration THEN  
41      SET var_interlude =  
42      get_datetime_variables_from_id(var_interlude_dv);  
43  
44      SET var_new_execution_time = NOW() +  
45      var_interlude;  
46  
47      UPDATE villages_resources_schedules  
48      SET execution_time = var_new_execution_time  
49      WHERE idvas = var_idvas;  
50  END IF;  
51  
52  END LOOP sync_process;  
53  
54  CLOSE curs;
```

55	
56	SELESAI

Penjelasan dari *procedure villages_resources_scheduling* pada tabel 4.57 yaitu:

1. *Procedure* akan dipanggil secara rutin dan terjadwal melalui fitur *event*.
2. Data jadwal penghasilan sumber daya yang diproses pada *procedure* ini ialah data yang memenuhi syarat yang telah ditetapkan pada algoritma SELECT dalam variabel *curs* di deklarasi awal.
3. Syarat yang harus dipenuhi ialah data dengan status tidak sama dengan “*iteration_done*” dan waktu penghasilan sumber daya harus lebih kecil sama dengan waktu pemrosesan.
4. Algoritma SELECT pada variabel *curs* menggunakan fitur FOR UPDATE yang berfungsi untuk menutup akses SELECT yang lain terhadap data yang telah atau sedang diproses oleh SELECT sebelumnya.
5. Deklarasi awal *curs* digunakan untuk mendapatkan keseluruhan data yang memenuhi syarat pada tabel, kemudian pada perulangan di baris 5-52 dilakukan proses penghasilan sumber daya pada desa pemain.

4.3.2.35 Implementasi *Procedure villages_troops_attacks_scheduling*

Procedure yang digunakan *administrator*, pengembang atau sistem untuk memproses data penyerangan pasukan pemain yang terdapat pada tabel *villages_troops_attacks_schedules*.

Tabel 4.58 Pseudocode *Procedure villages_troops_attacks_scheduling*

Pseudocode <i>Procedure villages_troops_attacks_scheduling</i>
Deklarasi awal <pre> var_idvtas:INT var_village_troop_attack:INT var_target_village:INT var_execution_time:TIMESTAMP </pre>


```
var_status_vv:INT
var_status_req:INT = get_varchar_variables_id("troop_home")
var_finished:INT = 0
var_status_done:INT
var_attack:INT
var_defense:INT
var_survival:INT
var_point:INT
var_min_point:INT
var_max_point:INT
var_attack_point:INT
var_defense_point:INT
var_defense_troop:VARCHAR(50)
var_attack_troop:VARCHAR(50)
var_temp_troop:VARCHAR(50)
var_temp_resource:VARCHAR(50)
var_target_village_resource:VARCHAR(50)
var_process:VARCHAR(50)
var_next_process:VARCHAR(50)
var_marker_a:INT
var_marker_k:INT
var_troop_type:INT
var_troop_amount:INT
var_victim:VARCHAR(50)
var_victim_type:INT
var_victim_amount:INT
var_survivor:VARCHAR(50)
var_survivor_type:INT
var_survivor_amount:INT
var_speedless_troop:INT
var_speedless_troop_speed:INT
var_village:INT
var_survivor_carrying_capacity:INT
var_resource_type:INT
var_resource_amount:INT
var_war_loot:VARCHAR(50)
var_war_loot_type:INT
var_war_loot_amount:INT
var_travel_time:TIME
var_status_dead:INT =
get_varchar_variables_id("troop_dead")
var_status_home:INT =
get_varchar_variables_id("troop_home")
var_status_back:INT =
get_varchar_variables_id("troop_back")
var_iteration_once:INT = get_varchar_variables_id
("iteration once")
```

```
var_idvt:INT
curs:CURSOR FOR
    SELECT idvtas,village_troop_attack,target_village,
    execution_time,status_vv
    FROM villages_troops_attacks_schedules
    WHERE execution_time <= NOW() AND status_vv !=
    var_status_req FOR UPDATE
CONTINUE HANDLER FOR NOT FOUND SET var_finished = 1;
```

Deskripsi

1	MULAI
2	
3	OPEN curs;
4	
5	sync_process: LOOP
6	
7	FETCH curs INTO var_idvtas,var_village_troop_attack,
8	var_target_village,var_execution_time,var_status_vv;
9	
10	IF var_finished = 1 THEN
11	LEAVE sync_process;
12	END IF;
13	
14	SELECT SUM(vt.value * iv.value) INTO var_attack
15	FROM villages_troops_attacks vta
16	LEFT JOIN villages_troops vt ON vt.idvt =
17	vta.village_troop
18	LEFT JOIN troops_stats ts ON ts.troop = vt.troop
19	LEFT JOIN mem_integer_variables iv ON iv.idiv =
20	ts.value_iv
21	LEFT JOIN attributes a ON a.ida = ts.attribute
22	WHERE a.attribute = "attack_point" AND
23	vta.attack = var_village_troop_attack;
24	
25	SELECT IFNULL(SUM(vt.value*iv.value),0) INTO
26	var_defense
27	FROM villages_troops vt
28	LEFT JOIN troops_stats ts ON ts.troop = vt.troop
29	LEFT JOIN mem_integer_variables iv ON iv.idiv =
30	ts.value_iv
31	LEFT JOIN attributes a ON a.ida = ts.attribute
32	WHERE a.attribute = "defense_point" AND vt.village =
33	var_target_village;
34	
35	SET var_survival = var_attack + var_defense;
36	

```

37 SELECT GROUP_CONCAT(CONCAT(at.troop,"#",at.amount)
38 SEPARATOR "&")
39 INTO var_attack_troop
40 FROM
41 (SELECT vt.troop,SUM(vt.value)*-1 AS amount
42 FROM villages_troops_attacks vta
43 LEFT JOIN villages_troops vt ON vta.village_troop =
44 vt.idvt
45 WHERE vta.attack = var_village_troop_attack
46 GROUP BY vt.troop) AS at;
47
48 SELECT GROUP_CONCAT(CONCAT(vt.troop,"#",vt.amount)
49 SEPARATOR "&")
50 INTO var_defense_troop
51 FROM
52 (SELECT troop,SUM(value) AS amount
53 FROM villages_troops
54 WHERE village = var_target_village
55 GROUP BY troop) AS vt;
56
57 IF var_survival < 0 THEN
58
59     SELECT
60     IF((-1*var_survival)-MIN(iv.value)<0,0,
61     (-1*var_survival)-MIN(iv.value)),
62     IF((-1*var_survival)+MIN(iv.value)<0,0,
63     (-1*var_survival)+MIN(iv.value))
64     INTO var_min_point,var_max_point
65     FROM villages_troops_attacks vta
66     LEFT JOIN villages_troops vt ON
67     vta.village_troop = vt.idvt
68     LEFT JOIN troops_stats ts ON ts.troop =
69     vt.troop
70     LEFT JOIN mem_integer_variables iv ON iv.idiv =
71     ts.value_iv
72     LEFT JOIN attributes a ON a.ida = ts.attribute
73     WHERE a.attribute = "attack_point" AND
74     vta.attack = var_village_troop_attack;
75
76     SET var_temp_troop = var_attack_troop;
77
78     REPEAT
79
80         SET var_point = 0;
81
82         SET var_survivor = NULL;
83

```



```

84      SET var_attack_troop = var_temp_troop;
85      REPEAT
86
87          SET var_marker_a =
88          LOCATE("&",var_attack_troop);
89
90          IF var_marker_a != 0 THEN
91              SET var_process = SUBSTRING(var_
92              attack_troop,1,var_marker_a -
93              1);
94
95              SET var_next_process = SUBSTRING
96              (var_attack_troop,var_marker_a +
97              1);
98          ELSE
99              SET var_process = SUBSTRING
100             (var_attack
101             _troop,var_marker_a + 1);
102          END IF;
103
104          SET var_attack_troop =
105          var_next_process;
106
107          SET var_marker_k = LOCATE
108          ("#",var_process);
109
110          SET var_troop_type = SUBSTRING
111          (var_process,1,var_marker_k - 1);
112
113          SET var_troop_amount = SUBSTRING(var_
114          process,var_marker_k + 1);
115
116          SET var_survivor_amount =
117          random_integer
118          (0,var_troop_amount);
119
120          SELECT iv.value INTO var_attack_point
121          FROM troops_stats ts
122          LEFT JOIN mem_integer_variables iv ON
123          ts.value_iv = iv.idiv
124          LEFT JOIN attributes a ON
125          ts.attribute = a.ida
126          WHERE a.attribute = "attack_point"
127          AND
128          troop = var_troop_type;
129
130          SET var point = var point +

```

```

131      (var_survivor_
132      amount * var_attack_point);
133
134      IF var_survivor IS NULL THEN
135          SET var_survivor =
136
137          CONCAT(var_troop_type,"#",
138          var_survivor_amount);
139      ELSE
140          SET var_survivor =
141          CONCAT(var_survivor,"&",
142          var_troop_type,"#",
143          var_survivor_amount);
144      END IF;
145
146      UNTIL var_marker_a = 0 END REPEAT;
147
148      UNTIL var_point >= var_min_point AND var_point
149      <=
150      var_max_point END REPEAT;
151
152      SET var_attack_troop := var_temp_troop;
153
154      SELECT
155      vt.troop,vt.village,MIN(iv.value)
156      INTO
157      var_speedless_troop,var_village,var_speedless
158      _troop_speed
159      FROM villages_troops_attacks vta
160      LEFT JOIN villages_troops vt ON
161      vta.village_troop = vt.idvt
162      LEFT JOIN troops_stats ts ON ts.troop =
163      vt.troop
164      LEFT JOIN mem_integer_variables iv ON iv.idiv =
165      ts.value_iv
166      LEFT JOIN attributes a ON a.ida = ts.attribute
167      WHERE a.attribute = "troop_speed" AND
168      vta.attack = var_village_troop_attack;
169
170      SET var_travel_time =
171      ADDTIME(NOW(),get_travel_time(var_speedless_tro
172      op,var_village,var_target_village));
173
174      SET var_survivor_carrying_capacity = 0;
175
176      SET var_temp_troop = var_survivor;
177

```

```

178      REPEAT
179          SET var_marker_a =
180          LOCATE("&",var_survivor);
181
182          IF var_marker_a != 0 THEN
183              SET var_process =
184              SUBSTRING(var_survivor,1,var_marker_a
185              - 1);
186
187              SET var_next_process =
188              SUBSTRING(var_survivor,var_marker_a +
189              1);
190          ELSE
191              SET var_process =
192              SUBSTRING(var_survivor,var_marker_a +
193              1);
194          END IF;
195
196          SET var_survivor = var_next_process;
197
198          SET var_marker_k =
199          LOCATE("#",var_process);
200
201          SET var_survivor_type =
202          SUBSTRING(var_process,1,var_marker_k - 1);
203
204          SET var_survivor_amount =
205          SUBSTRING(var_process,var_marker_k + 1);
206
207          SELECT
208          var_survivor_carrying_capacity +
209          (iv.value*var_survivor_amount)
210          INTO var_survivor_carrying_capacity
211          FROM troops_stats ts
212          LEFT JOIN attributes a ON ts.attribute =
213          a.ida
214          LEFT JOIN mem_integer_variables iv
215          ON ts.value_iv = iv.idiv
216          WHERE a.attribute = "carrying_capacity"
217          AND
218          ts.troop = var_survivor_type;
219
220          INSERT INTO
221          villages_troops_builds_schedules
222          SET
223          village = var_village,
224          troop = var_survivor_type,

```



```

225         value = var_survivor_amount,
226         finished = var_travel_time,
227         status_vv = var_status_back;
228
229     UNTIL var_marker_a = 0 END REPEAT;
230
231     SET var_survivor = var_temp_troop;
232
233     SELECT GROUP_CONCAT
234     (CONCAT(vr.attribute,"#",vr.amount)
235     SEPARATOR "&")
236     INTO var_target_village_resource
237     FROM
238     (
239         SELECT village,attribute,SUM(value) AS
240         amount
241         FROM villages_resources
242         WHERE village = var_target_village
243         GROUP BY attribute
244     ) AS vr;
245
246     SET var_temp_resource =
247     var_target_village_resource;
248
249     REPEAT
250
251         SET var_point = 0;
252
253         SET var_war_loot = NULL;
254
255         SET var_target_village_resource =
256         var_temp_resource;
257
258         REPEAT
259
260             SET var_marker_a =
261             LOCATE
262             ("&",var_target_village_resource);
263
264             IF var_marker_a != 0 THEN
265                 SET var_process =
266                 SUBSTRING
267                 (var_target_village_resource,
268                 1,var_marker_a - 1);
269
270                 SET var_next_process =
271                 SUBSTRING

```

```

272         (var_target_village_resource,
273         var_marker_a + 1);
274     ELSE
275         SET var_process =
276         SUBSTRING
277         (var_target_village_resource,
278         var_marker_a + 1);
279     END IF;
280
281     SET var_target_village_resource =
282     var_next_process;
283
284     SET var_marker_k = LOCATE
285     ("#",var_process);
286
287     SET var_resource_type =
288     SUBSTRING(var_process,1,var_marker_k
289     - 1);
290
291     SET var_resource_amount =
292     SUBSTRING(var_process,var_marker_k +
293     1);
294
295     SET var_war_loot_type =
296     var_resource_type;
297
298     IF(var_resource_amount <=
299     var_survivor_carrying_capacity) THEN
300         SET var_war_loot_amount =
301         random_integer
302         (0,var_resource_amount);
303     ELSE
304         SET var_war_loot_amount =
305         random_integer
306         (0,var_survivor
307         carrying_capacity);
308     END IF;
309
310     SET var_point = var_point +
311     var_war_loot_amount;
312
313     IF var_war_loot IS NULL THEN
314         SET var_war_loot =
315         CONCAT(var_war_loot_type,"#",
316         var_war_loot_amount);
317     ELSE
318         SET var_war_loot =

```

```

319         CONCAT(var_war_loot,"&",
320             var_war_loot_type,
321             "#",var_war_loot_amount);
322     END IF;
323
324     UNTIL var_marker_a = 0 END REPEAT;
325
326     UNTIL var_point <=
327     var_survivor_carrying_capacity END REPEAT;
328
329     SET var_target_village_resource =
330     var_temp_resource;
331
332     SET var_temp_resource = var_war_loot;
333
334     REPEAT
335
336         SET var_marker_a = LOCATE
337         ("&",var_war_loot);
338
339         IF var_marker_a != 0 THEN
340             SET var_process = SUBSTRING
341             (var_war_loot,1,var_marker_a - 1);
342
343             SET var_next_process = SUBSTRING
344             (var_war_loot,var_marker_a + 1);
345         ELSE
346             SET var_process = SUBSTRING
347             (var_war_loot,var_marker_a + 1);
348         END IF;
349
350         SET var_war_loot = var_next_process;
351
352         SET var_marker_k = LOCATE
353         ("#",var_process);
354
355         SET var_war_loot_type = SUBSTRING
356         (var_process,1,var_marker_k - 1);
357
358         SET var_war_loot_amount = SUBSTRING
359         (var_process,var_marker_k + 1);
360
361         INSERT INTO wars_loots_schedules
362         SET
363         village = var_village,
364         attribute = var_war_loot_type,
365         value = var war loot amount,

```



```

366         execution_time = var_travel_time,
367         status_vv = var_iteration_once;
368
369         INSERT INTO villages_resources
370         SET
371         village = var_target_village,
372         attribute = var_war_loot_type,
373         value = -1*var_war_loot_amount,
374         execution_time = var_travel_time;
375
376     UNTIL var_marker_a = 0 END REPEAT;
377
378     SET var_war_loot = var_temp_resource;
379
380     SET SQL_SAFE_UPDATES=0;
381
382     UPDATE villages_troops vt
383     LEFT JOIN villages_troops_attacks vta ON
384     vta.village_troop = vt.idvt
385     SET vt.status_vv = var_status_dead
386     WHERE vta.attack = var_village_troop_attack;
387
388     SET SQL_SAFE_UPDATES=1;
389
390     IF var_defense != 0 THEN
391     REPEAT
392
393         SET var_marker_a = LOCATE
394         ("&",var_defense_troop);
395
396         IF var_marker_a != 0 THEN
397             SET var_process = SUBSTRING
398             (var_defense_troop,1,var_marker_a -
399             1);
400
401             SET var_next_process = SUBSTRING
402             (var_defense_troop,var_marker_a + 1);
403         ELSE
404             SET var_process = SUBSTRING
405             (var_defense_troop,var_marker_a + 1);
406         END IF;
407
408         SET var_defense_troop = var_next_process;
409
410         SET var_marker_k = LOCATE
411         ("#",var_process);
412

```

```

413      SET var_troop_type = SUBSTRING
414      (var_process,1,var_marker_k - 1);
415
416      SET var_troop_amount = SUBSTRING
417      (var_process,var_marker_k + 1);
418
419      SET var_idvt = get_new_idvt();
420      INSERT INTO villages_troops
421      SET
422      idvt = var_idvt,
423      village = var_target_village,
424      troop = var_troop_type,
425      value = var_troop_amount * -1,
426      execution_time = var_execution_time,
427      status_vv = var_status_dead;
428
429      UNTIL var_marker_a = 0 END REPEAT;
430
431      END IF;
432
433      ELSEIF var_survival > 0 THEN
434
435          SELECT
436          var_defense-(var_survival-MIN(iv.value)),
437          var_defense-(var_survival+MIN(iv.value))
438          INTO var_max_point,var_min_point
439          FROM villages_troops vt
440          LEFT JOIN troops_stats ts ON ts.troop =
441          vt.troop
442          LEFT JOIN mem_integer_variables iv ON iv.idiv =
443          ts.value_iv
444          LEFT JOIN attributes a ON a.ida = ts.attribute
445          WHERE a.attribute = "defense_point" AND
446          vt.village =
447          var_target_village;
448
449          SET var_temp_troop = var_defense_troop;
450
451          REPEAT
452
453              SET var_point = 0;
454
455              SET var_victim = NULL;
456
457              SET var_defense_troop = var_temp_troop;
458
459              REPEAT

```

```

460
461      SET var_marker_a =
462      LOCATE("&",var_defense_troop);
463
464      IF var_marker_a != 0 THEN
465
466          SET var_process = SUBSTRING
467          (var_defense_troop,1,
468          var_marker_a - 1);
469
470          SET var_next_process = SUBSTRING
471          (var_defense_troop,var_marker_a
472          + 1);
473
474      ELSE
475
476          SET var_process = SUBSTRING
477          (var_defense_troop,var_marker_a
478          + 1);
479
480      END IF;
481
482      SET var_defense_troop =
483      var_next_process;
484
485      SET var_marker_k = LOCATE
486      ("#",var_process);
487
488      SET var_troop_type = SUBSTRING
489      (var_process,1,var_marker_k - 1);
490
491      SET var_troop_amount = SUBSTRING
492      (var_process,var_marker_k + 1);
493
494      SET var_victim_amount =
495      random_integer
496      (0,var_troop_amount);
497
498      SELECT iv.value INTO
499      var_defense_point
500      FROM troops_stats ts
501      LEFT JOIN mem_integer_variables iv ON
502      ts.value_iv = iv.idiv
503      LEFT JOIN attributes a ON
504      ts.attribute = a.ida
505      WHERE a.attribute = "defense_point"
506      AND troop = var_troop_type;

```



```
507
508         SET var_point = var_point +
509         (var_victim_amount *
510         var_defense_point);
511
512         IF var_victim IS NULL THEN
513             SET var_victim = CONCAT
514             (var_troop_type,"#",
515             var_victim_amount);
516         ELSE
517             SET var_victim = CONCAT
518             (var_victim,"&",
519             var_troop_type,
520             "#",var_victim_amount);
521         END IF;
522
523         UNTIL var_marker_a = 0 END REPEAT;
524
525     UNTIL var_point >= var_min_point AND var_point
526     <= var_max_point END REPEAT;
527
528     REPEAT
529
530         SET var_marker_a = LOCATE("&",var_victim);
531
532         IF var_marker_a != 0 THEN
533
534             SET var_process = SUBSTRING
535             (var_victim,1,var_marker_a - 1);
536
537             SET var_next_process = SUBSTRING
538             (var_victim,var_marker_a + 1);
539         ELSE
540             SET var_process = SUBSTRING
541             (var_victim,var_marker_a + 1);
542         END IF;
543
544         SET var_victim = var_next_process;
545
546         SET var_marker_k = LOCATE
547         ("#",var_process);
548
549         SET var_victim_type = SUBSTRING
550         (var_process,1,var_marker_k - 1);
551
552         SET var_victim_amount = SUBSTRING
553         (var_process,var_marker_k + 1);
```

```

554
555         SET var_idvt = get_new_idvt();
556
557         INSERT INTO villages_troops
558         SET
559         idvt = var_idvt,
560         village = var_target_village,
561         troop = var_victim_type,
562         value = var_victim_amount * -1,
563         execution_time = var_execution_time,
564         status_vv = var_status_dead;
565
566         UNTIL var_marker_a = 0 END REPEAT;
567
568         SET SQL_SAFE_UPDATES=0;
569
570         UPDATE villages_troops vt
571         LEFT JOIN villages_troops_attacks vta ON
572         vta.village_troop = vt.idvt
573         SET vt.status_vv = var_status_dead
574         WHERE vta.attack = var_village_troop_attack;
575
576         SET SQL_SAFE_UPDATES=1;
577
578     ELSE
579
580         SET SQL_SAFE_UPDATES=0;
581
582         UPDATE villages_troops vt
583         LEFT JOIN villages_troops_attacks vta ON
584         vta.village_troop = vt.idvt
585         SET vt.status_vv = var_status_dead
586         WHERE vta.attack = var_village_troop_attack;
587
588         SET SQL_SAFE_UPDATES=1;
589
590         REPEAT
591
592             SET var_marker_a = LOCATE
593             ("&",var_defense_troop);
594
595             IF var_marker_a != 0 THEN
596                 SET var_process = SUBSTRING
597                 (var_defense_troop,1,var_marker_a -
598                 1);
599
600                 SET var_next_process = SUBSTRING

```

```
601         (var_defense_troop,var_marker_a + 1);
602     ELSE
603         SET var_process = SUBSTRING
604         (var_defense_troop,var_marker_a + 1);
605     END IF;
606
607     SET var_defense_troop = var_next_process;
608
609     SET var_marker_k = LOCATE
610     ("#",var_process);
611
612     SET var_troop_type = SUBSTRING
613     (var_process,1,var_marker_k - 1);
614
615     SET var_troop_amount = SUBSTRING
616     (var_process,var_marker_k + 1);
617
618     SET var_idvt = get_new_idvt();
619
620     INSERT INTO villages_troops
621     SET
622     idvt = var_idvt,
623     village = var_target_village,
624     troop = var_troop_type,
625     value = var_troop_amount * -1,
626     execution_time = var_execution_time,
627     status_vv = var_status_dead;
628
629     UNTIL var_marker_a = 0 END REPEAT;
630
631 END IF;
632
633 SET SQL_SAFE_UPDATES=0;
634
635 UPDATE villages_troops_attacks_schedules
636 SET
637 status_vv = var_status_home
638 WHERE idvtas = var_idvtas;
639
640 SET SQL_SAFE_UPDATES=1;
641
642 END LOOP sync_process;
643
644 CLOSE curs;
645
646 SELESAI
647
```


Penjelasan dari *procedure* *villages_troops_attacks_scheduling* pada tabel 4.58 yaitu:

1. *Procedure* akan dipanggil secara rutin dan terjadwal melalui fitur *event*.
2. Data jadwal penyerangan pasukan yang diproses pada *procedure* ini ialah data yang memenuhi syarat yang telah ditetapkan pada algoritma SELECT dalam variabel *curs* di deklarasi awal.
3. Syarat yang harus dipenuhi ialah data dengan status tidak sama dengan “*iteration_done*” dan waktu penyerangan pasukan harus lebih kecil sama dengan waktu pemrosesan.
4. Algoritma SELECT pada variabel *curs* menggunakan fitur FOR UPDATE yang berfungsi untuk menutup akses SELECT yang lain terhadap data yang telah atau sedang diproses oleh SELECT sebelumnya.
5. Deklarasi awal *curs* digunakan untuk mendapatkan keseluruhan data yang memenuhi syarat pada tabel, kemudian pada perulangan di baris 5-642 dilakukan proses penyerangan pasukan.
6. Dikarenakan tidak adanya variabel *array* pada MySQL maka perlu dibuat mekanisme pengganti yang mampu menggantikan fungsi variabel *array* yang diperlukan dalam perangkat lunak.
7. Mekanisme pengganti yang dipilih ialah penggunaan variabel VARCHAR sebagai penampung kumpulan nilai yang dibutuhkan.
8. Nantinya kumpulan nilai dalam variabel VARCHAR akan diuraikan berdasarkan karakter tertentu yakni karakter ‘#’ dan karakter ‘&’.
9. Format pengkodean kumpulan nilai ialah :
“*jenis_pasukan#jumlah_pasukan&jenis_pasukan#jumlah_pasukan&...* ”.
10. Penguraian variabel pengganti menggunakan fitur perulangan, *function* LOCATE dan *function* SUBSTRING dalam MySQL seperti pada baris 5-642.

11. Untuk *generate* variabel *array* pada kumpulan nilai hasil dari SELECT digunakan kombinasi fungsi GROUP_CONCAT dan CONCAT seperti pada baris 233-234.
12. Seleksi kondisi pada baris 57-631 bertujuan untuk memproses hasil peperangan dari algoritma pada baris 14-35.
13. Terdapat 3 kondisi yang memungkinkan dari hasil peperangan yakni kemenangan pasukan penyerang, kekalahan pasukan penyerang serta hasil seri yang ditandai dengan gugurnya seluruh pasukan penyerang dan pasukan desa terserang.
14. Algoritma proses yang dilakukan pada kondisi kemenangan pasukan penyerang ialah menentukan jenis dan jumlah pasukan penyerang yang tersisa, menggugurkan seluruh pasukan desa terserang, menentukan jenis dan jumlah sumber daya rampasan perang serta menjadwalkan kepulauan sisa pasukan penyerang dan sumber daya.
15. Algoritma proses yang dilakukan pada kondisi kekalahan pasukan penyerang ialah menentukan jenis dan jumlah pasukan pertahanan desa yang tersisa menggugurkan seluruh pasukan penyerang.
16. Algoritma proses yang dilakukan pada kondisi seri ialah menggugurkan seluruh pasukan penyerang dan pasukan desa terserang.

4.3.2.36 Implementasi Procedure villages_troops_builds_scheduling

Procedure yang digunakan *administrator*, pengembang atau sistem untuk memproses data jadwal pelatihan pasukan pertahanan maupun penyerangan desa yang ada pada tabel villages_troops_builds_schedules.

Tabel 4.59 Pseudocode Procedure villages_troops_builds_scheduling

Pseudocode Procedure villages_troops_builds_scheduling
Deklarasi awal var_idvtbs:INT var_village:INT

```

var_troop:INT
var_value:INT
var_finished_build:TIMESTAMP
var_status_vv:INT
var_status_req:INT =
get_varchar_variables_id("iteration_done")
var_finished:INT = 0
var_status_done:INT
var_status_home:INT
var_idvt:INT
curs:CURSOR FOR
    SELECT idvtbs,village,troop,value,finished,status_vv
    FROM villages_troops_builds_schedules
    WHERE finished <= NOW() AND status_vv !=
    var_status_req FOR UPDATE
CONTINUE HANDLER FOR NOT FOUND SET var_finished = 1

```

Deskripsi

1	MULAI
2	
3	OPEN curs;
4	
5	sync_process: LOOP
6	
7	FETCH curs INTO
8	var_idvtbs,var_village,var_troop,var_value,
9	var_finished_build,var_status_vv;
10	
11	IF var_finished = 1 THEN
12	
13	LEAVE sync_process;
14	
15	END IF;
16	
17	SET var_idvt = get_new_idvt();
18	
19	SET var_status_home =
20	get_varchar_variables_id("troop_home");
21	
22	INSERT INTO villages_troops
23	SET
24	idvt = var_idvt,
25	village = var_village,
26	troop = var_troop,
27	value = var_value,
28	execution time = var finished build,


```
29 status_vv = var_status_home;
30
31 SET var_status_done =
32 get_varchar_variables_id("iteration_done");
33
34 UPDATE villages_troops_builds_schedules
35 SET status_vv = var_status_done
36 WHERE idvtbs = var_idvtbs;
37
38 END LOOP sync_process;
39
40 CLOSE curs;
41
42 SELESAI
```

Penjelasan dari *procedure* *villages_troops_builds_scheduling* pada tabel 4.59 yaitu:

1. *Procedure* akan dipanggil secara rutin dan terjadwal melalui fitur *event*.
2. Data jadwal pelatihan pasukan yang diproses pada *procedure* ini ialah data yang memenuhi syarat yang telah ditetapkan pada algoritma SELECT dalam variabel *curs* di deklarasi awal.
3. Syarat yang harus dipenuhi ialah data dengan status tidak sama dengan "iteration_done" dan waktu penyelesaian pelatihan lebih kecil sama dengan waktu pemrosesan.
4. Algoritma SELECT pada variabel *curs* menggunakan fitur FOR UPDATE yang berfungsi untuk menutup akses SELECT yang lain terhadap data yang telah atau sedang diproses oleh SELECT sebelumnya.
5. Deklarasi awal *curs* digunakan untuk mendapatkan keseluruhan data yang memenuhi syarat pada tabel, kemudian pada perulangan di baris 5-38 dilakukan proses penyelesaian pelatihan pasukan pada desa pemain.

4.3.2.37 Implementasi Procedure wars_loots_scheduling

Procedure yang digunakan *administrator*, pengembang atau sistem untuk memproses data jadwal datangnya sumber daya rampasan yang ada pada tabel wars_loots_schedules.

Tabel 4.60 Pseudocode Procedure wars_loots_scheduling

Pseudocode Procedure wars_loots_scheduling	
Deklarasi awal <pre> var_idwls:INT var_village:INT var_attribute:INT var_value:INT var_execution_time:TIMESTAMP var_status_vv:INT var_status_req:INT = get_varchar_variables_id("iteration_done") var_status_done:INT = get_varchar_variables_id("iteration_done") var_finished INT DEFAULT 0; curs:CURSOR FOR SELECT idwls,village,attribute,value,execution_time, status_vv FROM wars_loots_schedules WHERE execution_time <= NOW() AND status_vv != var_status_req FOR UPDATE CONTINUE HANDLER FOR NOT FOUND SET var_finished = 1 </pre> Deskripsi	
1	MULAI
2	
3	OPEN curs;
4	
5	sync_process: LOOP
6	
7	FETCH curs INTO
8	var_idwls,var_village,var_attribute,var_value,
9	var_execution_time,var_status_vv;
10	
11	IF var_finished = 1 THEN

```
12
13     LEAVE sync_process;
14
15 END IF;
16
17 INSERT INTO villages_resources
18 SET
19 village = var_village,
20 attribute = var_attribute,
21 value = var_value,
22 execution_time = var_execution_time;
23
24 UPDATE wars_loots_schedules
25 SET status_vv = var_status_done
26 WHERE idwls = var_idwls;
27
28 END LOOP sync_process;
29
30 CLOSE curs;
31
32 SELESAI
```

Penjelasan dari *procedure* wars_loots_scheduling pada tabel 4.60 yaitu:

1. *Procedure* akan dipanggil secara rutin dan terjadwal melalui fitur *event*.
2. Data jadwal datangnya sumber daya rampasan yang diproses pada *procedure* ini ialah data yang memenuhi syarat yang telah ditetapkan pada algoritma SELECT dalam variabel *curs* di deklarasi awal.
3. Syarat yang harus dipenuhi ialah data dengan status tidak sama dengan “iteration_done” dan waktu datangnya sumber daya rampasan harus lebih kecil sama dengan waktu pemrosesan.
4. Algoritma SELECT pada variabel *curs* menggunakan fitur FOR UPDATE yang berfungsi untuk menutup akses SELECT yang lain terhadap data yang telah atau sedang diproses oleh SELECT sebelumnya.
5. Deklarasi awal *curs* digunakan untuk mendapatkan keseluruhan data yang memenuhi syarat pada tabel, kemudian pada perulangan di baris 5-28 dilakukan proses datangnya sumber daya rampasan di desa pemain.

4.3.2.38 Implementasi Procedure web_building_prices

Procedure yang dipakai untuk menampilkan data jenis serta sumber daya yang diperlukan untuk membangun tiap jenis bangunan di desa pemain.

Tabel 4.61 Pseudocode Procedure web_building_prices

Pseudocode Procedure web_building_prices	
Deklarasi awal	
Deskripsi	
1	MULAI
2	
3	SELECT * FROM view_buildings_prices;
4	
5	SELESAI

Penjelasan dari *procedure* web_building_prices pada tabel 4.61 yaitu:

1. Merupakan salah satu *procedure* yang boleh diakses langsung oleh aplikasi *web*.
2. Aplikasi *web* hanya diperkenankan mengakses *procedure* dengan nama yang diawali “web”.
3. *Procedure* tidak diperkenankan mengakses secara langsung tabel-tabel yang ada dan hanya boleh mengakses *procedure* lain atau *view*.

4.3.2.39 Implementasi Procedure web_count_enemy_attack_schedules

Procedure yang dipakai untuk menampilkan jumlah serangan pasukan desa lain ke desa pemain.

Tabel 4.62 Pseudocode Procedure web_count_enemy_attack_schedules

Pseudocode Procedure web_count_enemy_attack_schedules	
Deklarasi awal	

var_village:INT -> Masukan dari user/sistem	
Deskripsi	
1	MULAI
2	
3	SELECT COUNT(*) AS amount FROM
4	view_villages_troops_attacks_schedules
5	WHERE target_village = var_village
6	AND status = "away";
7	
8	SELESAI

Penjelasan dari *procedure* web_count_enemy_attack_schedules pada tabel 4.62 yaitu:

1. Merupakan salah satu *procedure* yang boleh diakses langsung oleh aplikasi web.
2. Aplikasi web hanya diperkenankan mengakses *procedure* dengan nama yang diawali "web".
3. *Procedure* tidak diperkenankan mengakses secara langsung tabel-tabel yang ada dan hanya boleh mengakses *procedure* lain atau view.
4. Diperlukan masukan berupa ID desa.

4.3.2.40 Implementasi Procedure web_count_troop_attack_schedules

Procedure yang dipakai untuk menampilkan jumlah serangan pasukan desa pemain ke desa lain.

Tabel 4.63 Pseudocode Procedure web_count_troop_attack_schedules

Pseudocode Procedure web_count_troop_attack_schedules
Deklarasi awal
var_village:INT -> Masukan dari user/sistem

Deskripsi	
1	MULAI
2	
3	SELECT COUNT(*) AS amount FROM
4	view_villages_troops_attacks_schedules
5	WHERE village = var_village
6	AND status = "away";
7	
8	SELESAI

Penjelasan dari *procedure* web_count_troop_attack_schedules pada tabel 4.63 yaitu:

1. Merupakan salah satu *procedure* yang boleh diakses langsung oleh aplikasi web.
2. Aplikasi web hanya diperkenankan mengakses *procedure* dengan nama yang diawali "web".
3. *Procedure* tidak diperkenankan mengakses secara langsung tabel-tabel yang ada dan hanya boleh mengakses *procedure* lain atau view.
4. Diperlukan masukan berupa ID desa.

4.3.2.41 Implementasi Procedure web_count_troop_back_schedules

Procedure yang dipakai untuk menampilkan jumlah jadwal pasukan desa pemain yang kembali dari desa lain.

Tabel 4.64 Pseudocode Procedure web_count_troop_back_schedules

Pseudocode Procedure web_count_troop_back_schedules	
Deklarasi awal	
var_idvillage:INT -> Masukan dari user/sistem	
Deskripsi	
1	MULAI

2	
3	SELECT COUNT(*) AS amount
4	FROM view_villages_troops_builds_schedules
5	WHERE status = "back"
6	AND village = var_idvillage;
7	
8	SELESAI

Penjelasan dari *procedure* *web_count_troop_back_schedules* pada tabel 4.64 yaitu:

1. Merupakan salah satu *procedure* yang boleh diakses langsung oleh aplikasi *web*.
2. Aplikasi *web* hanya diperkenankan mengakses *procedure* dengan nama yang diawali "web".
3. *Procedure* tidak diperkenankan mengkases secara langsung tabel-tabel yang ada dan hanya boleh mengkases *procedure* lain atau *view*.
4. Diperlukan masukan berupa ID desa.

4.3.2.42 Implementasi Procedure *web_count_village_building_build_schedules*

Procedure yang dipakai untuk menampilkan jumlah jadwal pembangunan bangunan pada desa pemain.

Tabel 4.65 Pseudocode Procedure *web_count_village_building_build_schedules*

Pseudocode Procedure <i>web_count_village_building_build_schedules</i>	
Deklarasi awal	
var_village:INT -> Masukan dari user/sistem	
Deskripsi	
1	MULAI
2	
3	SELECT count(*) as amount

4	FROM view_villages_buildings_builds_schedules
5	WHERE village = var_village
6	AND status_vv = "once";
7	
8	SELESAI

Penjelasan dari *procedure* web_count_village_building_build_schedules pada tabel 4.65 yaitu:

1. Merupakan salah satu *procedure* yang boleh diakses langsung oleh aplikasi *web*.
2. Aplikasi *web* hanya diperkenankan mengakses *procedure* dengan nama yang diawali "web".
3. *Procedure* tidak diperkenankan mengakses secara langsung tabel-tabel yang ada dan hanya boleh mengakses *procedure* lain atau *view*.
4. Diperlukan masukan berupa ID desa.

4.3.2.43 Implementasi Procedure web_count_village_troop_build_schedules

Procedure yang dipakai untuk menampilkan jumlah jadwal pelatihan pasukan pada desa pemain.

Tabel 4.66 Pseudocode Procedure web_count_village_troop_build_schedules

Pseudocode Procedure web_count_village_troop_build_schedules	
Deklarasi awal	
var_idvillage:INT -> Masukan dari user/sistem	
Deskripsi	
1	MULAI
2	
3	SELECT COUNT(*) AS amount
4	FROM view_villages_troops_builds_schedules
5	WHERE village = var_idvillage
6	AND status = "once";

7	
8	SELESAI

Penjelasan dari *procedure* `web_count_village_troop_build_schedules` pada tabel 4.66 yaitu:

1. Merupakan salah satu *procedure* yang boleh diakses langsung oleh aplikasi *web*.
2. Aplikasi *web* hanya diperkenankan mengakses *procedure* dengan nama yang diawali "web".
3. *Procedure* tidak diperkenankan mengakses secara langsung tabel-tabel yang ada dan hanya boleh mengakses *procedure* lain atau *view*.
4. Diperlukan masukan berupa ID desa.

4.3.2.44 Implementasi Procedure `web_count_war_loot_schedules`

Procedure yang dipakai untuk menampilkan jumlah jadwal pelatihan pasukan pada desa pemain.

Tabel 4.67 Pseudocode Procedure `web_count_war_loot_schedules`

Pseudocode Procedure <code>web_count_war_loot_schedules</code>	
Deklarasi awal	
<code>var_idvillage:INT -> Masukan dari user/sistem</code>	
Deskripsi	
1	MULAI
2	
3	<code>SELECT COUNT(*) AS amount</code>
4	<code>FROM view_wars_loots_schedules</code>
5	<code>WHERE village = var_idvillage AND</code>
6	<code>STATUS = "once";</code>
7	
8	SELESAI

Penjelasan dari *procedure* `web_count_war_loot_schedules` pada tabel 4.67 yaitu:

1. Merupakan salah satu *procedure* yang boleh diakses langsung oleh aplikasi *web*.
2. Aplikasi *web* hanya diperkenankan mengakses *procedure* dengan nama yang diawali “web”.
3. *Procedure* tidak diperkenankan mengakses secara langsung tabel-tabel yang ada dan hanya boleh mengakses *procedure* lain atau *view*.
4. Diperlukan masukan berupa ID desa.

4.3.2.45 Implementasi *Procedure* `web_default_active_village`

Procedure yang dipakai untuk mendapatkan ID desa pertama yang dimiliki pemain.

Tabel 4.68 Pseudocode *Procedure* `web_default_active_village`

Pseudocode <i>Procedure</i> <code>web_default_active_village</code>	
Deklarasi awal	
<code>var_username:VARCHAR(50) -> Masukan dari user/sistem</code> <code>var_idv:INT -> Keluaran dari user/sistem</code>	
Deskripsi	
1	MULAI
2	
3	<code>SELECT idv INTO var_idv</code>
4	<code>FROM view_players_villages</code>
5	<code>WHERE username = var_username</code>
6	<code>LIMIT 1;</code>
7	
8	SELESAI

Penjelasan dari *procedure* `web_default_active_village` pada tabel 4.68 yaitu:

1. Merupakan salah satu *procedure* yang boleh diakses langsung oleh aplikasi *web*.
2. Aplikasi *web* hanya diperkenankan mengakses *procedure* dengan nama yang diawali “web”.
3. *Procedure* tidak diperkenankan mengakses secara langsung tabel-tabel yang ada dan hanya boleh mengakses *procedure* lain atau *view*.
4. Diperlukan masukan berupa *username* pemain.
5. Keluaran berupa ID desa.

4.3.2.46 Implementasi Procedure web_enemy_attack_schedules

Procedure yang dipakai untuk mendapatkan seluruh jadwal serangan pasukan desa lain ke desa pemain.

Tabel 4.69 Pseudocode Procedure web_enemy_attack_schedules

Pseudocode Procedure web_enemy_attack_schedules	
Deklarasi awal	
var_village:INT -> Masukan dari user/sistem	
Deskripsi	
1	MULAI
2	
3	SELECT * FROM view_villages_troops_attacks_schedules
4	WHERE target_village = var_village
5	AND status = "away";
6	
7	SELESAI

Penjelasan dari *procedure* web_enemy_attack_schedules pada tabel 4.69 yaitu:

1. Merupakan salah satu *procedure* yang boleh diakses langsung oleh aplikasi *web*.

2. Aplikasi *web* hanya diperkenankan mengakses *procedure* dengan nama yang diawali “web”.
3. *Procedure* tidak diperkenankan mengakses secara langsung tabel-tabel yang ada dan hanya boleh mengakses *procedure* lain atau *view*.
4. Diperlukan masukan berupa ID desa.

4.3.2.47 Implementasi Procedure web_log_in

Procedure yang dipakai untuk melakukan verifikasi *username* dan *password* pemain sebelum memasuki dunia permainan.

Tabel 4.70 Pseudocode Procedure web_log_in

Pseudocode Procedure web_log_in	
Deklarasi awal	
var_username:VARCHAR(50) -> Masukan dari user/sistem	
var_password:VARCHAR(50) -> Masukan dari user/sistem	
var_status:VARCHAR(50) -> Keluaran dari user/sistem	
Deskripsi	
1	MULAI
2	
3	SELECT IF (COUNT (username) = 0, "error", "success") INTO
4	var_status
5	FROM players
6	WHERE username = var_username
7	AND
8	password =var_password;
9	
10	SELESAI

Penjelasan dari *procedure* web_log_in pada tabel 4.70 yaitu:

1. Merupakan salah satu *procedure* yang boleh diakses langsung oleh aplikasi *web*.

2. Aplikasi *web* hanya diperkenankan mengakses *procedure* dengan nama yang diawali “web”.
3. *Procedure* tidak diperkenankan mengakses secara langsung tabel-tabel yang ada dan hanya boleh mengakses *procedure* lain atau *view*.
4. Diperlukan masukan berupa *username* dan *password* pemain.
5. Keluaran berupa status hasil verifikasi.

4.3.2.48 Implementasi Procedure web_order_attack

Procedure yang dipakai pemain untuk melakukan penyerangan terhadap desa lain.

Tabel 4.71 Pseudocode Procedure web_order_attack

Pseudocode Procedure web_order_attack	
Deklarasi awal var_idvillage:INT -> Masukan dari user/sistem var_troop:VARCHAR(50) -> Masukan dari user/sistem var_target_village:INT -> Masukan dari user/sistem Deskripsi	
1	MULAI
2	
3	CALL
4	order_attack(var_idvillage,var_troop,var_target_village);
5	
6	SELESAI

Penjelasan dari *procedure* web_order_attack pada tabel 4.71 yaitu:

1. Merupakan salah satu *procedure* yang boleh diakses langsung oleh aplikasi *web*.
2. Aplikasi *web* hanya diperkenankan mengakses *procedure* dengan nama yang diawali “web”.

3. *Procedure* tidak diperkenankan mengakses secara langsung tabel-tabel yang ada dan hanya boleh mengakses *procedure* lain atau *view*.
4. Diperlukan masukan berupa ID desa pemain, pasukan yang akan dikirim dan ID desa target yang akan diserang.
5. Format masukan pasukan yang akan dikirim ialah :
“jenis_pasukan#jumlah_pasukan&jenis_pasukan#jumlah_pasukan&... ”.

4.3.2.49 Implementasi *Procedure web_order_building*

Procedure yang dipakai pemain untuk melakukan pembangunan bangunan, baik bangunan penghasil sumber daya maupun bangunan non-penghasil sumber daya pada desa pemain.

Tabel 4.72 Pseudocode *Procedure web_order_building*

Pseudocode <i>Procedure web_order_building</i>	
Deklarasi awal	
var_village:INT -> Masukan dari user/sistem	
var_building:VARCHAR(50) -> Masukan dari user/sistem	
Deskripsi	
1	MULAI
2	
3	CALL order_building(var_village,var_building);
4	
5	SELESAI

Penjelasan dari *procedure web_order_building* pada tabel 4.72 yaitu:

1. Merupakan salah satu *procedure* yang boleh diakses langsung oleh aplikasi *web*.
2. Aplikasi *web* hanya diperkenankan mengakses *procedure* dengan nama yang diawali “web”.

3. *Procedure* tidak diperkenankan mengakses secara langsung tabel-tabel yang ada dan hanya boleh mengakses *procedure* lain atau *view*.
4. Diperlukan masukan berupa ID desa pemain dan nama bangunan yang ingin dibangun.

4.3.2.50 Implementasi *Procedure web_order_troop*

Procedure yang dipakai pemain untuk melakukan pelatihan pasukan pertahanan maupun pasukan penyerang pada desa pemain.

Tabel 4.73 Pseudocode *Procedure web_order_troop*

Pseudocode <i>Procedure web_order_troop</i>	
Deklarasi awal	
var_village:INT -> Masukan dari user/sistem	
var_troop:VARCHAR(50) -> Masukan dari user/sistem	
var_troop_amount:INT = 1	
Deskripsi	
1	MULAI
2	
3	CALL
4	order_troop(var_village,var_troop,var_troop_amount);
5	
6	SELESAI

Penjelasan dari *procedure web_order_troop* pada tabel 4.73 yaitu:

1. Merupakan salah satu *procedure* yang boleh diakses langsung oleh aplikasi *web*.
2. Aplikasi *web* hanya diperkenankan mengakses *procedure* dengan nama yang diawali “web”.
3. *Procedure* tidak diperkenankan mengakses secara langsung tabel-tabel yang ada dan hanya boleh mengakses *procedure* lain atau *view*.

4. Diperlukan masukan berupa ID desa pemain dan jenis pasukan yang ingin dilatih.

4.3.2.51 Implementasi Procedure web_sign_up

Procedure yang dipakai calon pemain untuk melakukan pendaftaran pada *game server online* sebelum dapat memulai permainan.

Tabel 4.74 Pseudocode Procedure web_sign_up

Pseudocode Procedure web_sign_up	
Deklarasi awal	
var_username:VARCHAR(50) -> Masukan dari user/sistem var_password:VARCHAR(50) -> Masukan dari user/sistem var_password_conf:VARCHAR(50) -> Masukan dari user/sistem var_status:VARCHAR(50) -> Keluaran dari user/sistem	
Deskripsi	
1	MULAI
2	
3	IF(var_password = var_password_conf) THEN
4	
5	CALL web_username_availability
6	(var_username,@var_status);
7	
8	IF(@var_status = "yes") THEN
9	
10	CALL player_registration
11	(var_username,var_password);
12	
13	SET var_status := "success";
14	
15	ELSE
16	
17	SET var_status := "error";
18	
19	END IF;
20	
21	ELSE
22	

23	SET var_status := "error";
24	
25	END IF;
26	
27	SELESAI

Penjelasan dari *procedure* web_sign_up pada tabel 4.74 yaitu:

1. Merupakan salah satu *procedure* yang boleh diakses langsung oleh aplikasi web.
2. Aplikasi web hanya diperkenankan mengakses *procedure* dengan nama yang diawali "web".
3. *Procedure* tidak diperkenankan mengakses secara langsung tabel-tabel yang ada dan hanya boleh mengakses *procedure* lain atau view.
4. Diperlukan masukan berupa *username*, *password* dan konfirmasi *password* pemain.
5. Keluaran berupa status pendaftaran.
6. Seleksi kondisi pada baris 3 bertujuan untuk melakukan verifikasi kesamaan antara masukan *password* dan masukan konfirmasi *password* pendaftar.
7. Seleksi kondisi pada baris 8 dilakukan untuk melihat ketersediaan *username* yang dipilih oleh pendaftar.

4.3.2.52 Implementasi Procedure web_troop_attack_schedules

Procedure yang dipakai untuk mendapatkan seluruh jadwal serangan pasukan desa pemain ke desa lain.

Tabel 4.75 Pseudocode Procedure web_troop_attack_schedules

Pseudocode Procedure web_troop_attack_schedules
Deklarasi awal
var_village:INT -> Masukan dari user/sistem
Deskripsi

1	MULAI
2	
3	SELECT * FROM view_villages_troops_attacks_schedules
4	WHERE village = var_village
5	AND status = "away";
6	
7	SELESAI

Penjelasan dari *procedure* web_troop_attack_schedules pada tabel 4.75 yaitu:

1. Merupakan salah satu *procedure* yang boleh diakses langsung oleh aplikasi *web*.
2. Aplikasi *web* hanya diperkenankan mengakses *procedure* dengan nama yang diawali "web".
3. *Procedure* tidak diperkenankan mengakses secara langsung tabel-tabel yang ada dan hanya boleh mengakses *procedure* lain atau *view*.
4. Diperlukan masukan berupa ID desa.

4.3.2.53 Implementasi Procedure web_troop_back_schedules

Procedure yang dipakai untuk mendapatkan seluruh jadwal kembalinya pasukan desa pemain dari desa lain.

Tabel 4.76 Pseudocode Procedure web_troop_back_schedules

Pseudocode Procedure web_troop_back_schedules	
Deklarasi awal	
var_idvillage:INT -> Masukan dari user/sistem	
Deskripsi	
1	MULAI
2	
3	SELECT *
4	FROM view_villages_troops_builds_schedules
5	WHERE status = "back"

6	AND village = var_idvillage;
7	
8	SELESAI

Penjelasan dari *procedure* web_troop_back_schedules pada tabel 4.76 yaitu:

1. Merupakan salah satu *procedure* yang boleh diakses langsung oleh aplikasi web.
2. Aplikasi web hanya diperkenankan mengakses *procedure* dengan nama yang diawali “web”.
3. *Procedure* tidak diperkenankan mengakses secara langsung tabel-tabel yang ada dan hanya boleh mengakses *procedure* lain atau view.
4. Diperlukan masukan berupa ID desa.

4.3.2.54 Implementasi Procedure web_troop_prices

Procedure yang dipakai untuk menampilkan data jenis serta sumber daya yang diperlukan untuk melatih tiap jenis pasukan di desa pemain.

Tabel 4.77 Pseudocode Procedure web_troop_prices

Pseudocode Procedure web_troop_prices	
Deklarasi awal	
Deskripsi	
1	MULAI
2	
3	SELECT * FROM view_troops_prices;
4	
5	SELESAI

Penjelasan dari *procedure* web_troop_prices pada tabel 4.77 yaitu:

1. Merupakan salah satu *procedure* yang boleh diakses langsung oleh aplikasi web.

2. Aplikasi *web* hanya diperkenankan mengakses *procedure* dengan nama yang diawali “web”.
3. *Procedure* tidak diperkenankan mengakses secara langsung tabel-tabel yang ada dan hanya boleh mengakses *procedure* lain atau *view*.

4.3.2.55 Implementasi *Procedure web_troop_stats*

Procedure yang dipakai untuk menampilkan data lengkap atribut tiap jenis pasukan yang bisa dilatih di desa pemain.

Tabel 4.78 Pseudocode *Procedure web_troop_stats*

Pseudocode <i>Procedure web_troop_stats</i>	
Deklarasi awal	
Deskripsi	
1	MULAI
2	
3	SELECT * FROM <i>view_troops_stats</i> ;
4	
5	SELESAI

Penjelasan dari *procedure web_troop_stats* pada tabel 4.78 yaitu:

1. Merupakan salah satu *procedure* yang boleh diakses langsung oleh aplikasi *web*.
2. Aplikasi *web* hanya diperkenankan mengakses *procedure* dengan nama yang diawali “web”.
3. *Procedure* tidak diperkenankan mengakses secara langsung tabel-tabel yang ada dan hanya boleh mengakses *procedure* lain atau *view*.

4.3.2.56 Implementasi Procedure web_username_availability

Procedure yang dipakai untuk melakukan verifikasi ketersediaan *username* yang dipilih calon pemain ketika proses pendaftaran.

Tabel 4.79 Pseudocode Procedure web_username_availability

Pseudocode Procedure web_username_availability	
Deklarasi awal	
var_username:VARCHAR(50) -> Masukan dari user/sistem	
var_availability:VARCHAR(10) -> Keluaran dari user/sistem	
Deskripsi	
1	MULAI
2	
3	SELECT IF(COUNT(username)=0,"yes","no") INTO
4	var_availability
5	FROM players
6	WHERE username = var_username;
7	
8	SELESAI

Penjelasan dari *procedure* web_username_availability pada tabel 4.79 yaitu:

1. Merupakan salah satu *procedure* yang boleh diakses langsung oleh aplikasi *web*.
2. Aplikasi *web* hanya diperkenankan mengakses *procedure* dengan nama yang diawali "web".
3. *Procedure* tidak diperkenankan mengakses secara langsung tabel-tabel yang ada dan hanya boleh mengakses *procedure* lain atau *view*.
4. Diperlukan masukan berupa *username* pemain.
5. Keluaran berupa status hasil verifikasi.

4.3.2.57 Implementasi Procedure web_village_building_amounts

Procedure yang dipakai untuk menampilkan bangunan dengan jenis tertentu yang dimiliki desa pemain lengkap beserta jumlah bangunannya.

Tabel 4.80 Pseudocode Procedure web_village_building_amounts

Pseudocode Procedure web_village_building_amounts	
Deklarasi awal	
var_village:INT -> Masukan dari user/sistem	
var_building:VARCHAR(50) -> Masukan dari user/sistem	
Deskripsi	
1	MULAI
2	
3	SELECT *
4	FROM view_villages_buildings
5	WHERE village = var_village
6	AND building = var_building;
7	
8	SELESAI

Penjelasan dari *procedure* web_village_building_amounts pada tabel 4.80 yaitu:

1. Merupakan salah satu *procedure* yang boleh diakses langsung oleh aplikasi *web*.
2. Aplikasi *web* hanya diperkenankan mengakses *procedure* dengan nama yang diawali “web”.
3. *Procedure* tidak diperkenankan mengakses secara langsung tabel-tabel yang ada dan hanya boleh mengakses *procedure* lain atau *view*.
4. Diperlukan masukan berupa ID desa pemain dan jenis bangunan yang ingin diketahui jumlahnya.

4.3.2.58 Implementasi Procedure web_village_building_build_schedules

Procedure yang dipakai untuk mendapatkan seluruh jadwal pembangunan bangunan pada desa pemain.

Tabel 4.81 Pseudocode Procedure web_village_building_build_schedules

Pseudocode Procedure web_village_building_build_schedules	
Deklarasi awal	
var_idvillage:INT -> Masukan dari user/sistem	
Deskripsi	
1	MULAI
2	
3	SELECT *
4	FROM view_villages_buildings_builds_schedules
5	WHERE village = var_village
6	AND status_vv = "once";
7	
8	SELESAI

Penjelasan dari *procedure* web_village_building_build_schedules pada tabel 4.81 yaitu:

1. Merupakan salah satu *procedure* yang boleh diakses langsung oleh aplikasi *web*.
2. Aplikasi *web* hanya diperkenankan mengakses *procedure* dengan nama yang diawali "web".
3. *Procedure* tidak diperkenankan mengakses secara langsung tabel-tabel yang ada dan hanya boleh mengakses *procedure* lain atau *view*.
4. Diperlukan masukan berupa ID desa.

4.3.2.59 Implementasi Procedure web_village_buildings

Procedure yang dipakai untuk menampilkan seluruh jenis bangunan yang dimiliki desa pemain lengkap beserta jumlah bangunannya.

Tabel 4.82 Pseudocode Procedure web_village_buildings

Pseudocode Procedure web_village_buildings	
Deklarasi awal	
var_idvillage:INT -> Masukan dari user/sistem	
Deskripsi	
1	MULAI
2	
3	SELECT *
4	FROM view_villages_buildings
5	WHERE village = var_village;
6	
7	SELESAI

Penjelasan dari *procedure* web_village_buildings pada tabel 4.82 yaitu:

1. Merupakan salah satu *procedure* yang boleh diakses langsung oleh aplikasi *web*.
2. Aplikasi *web* hanya diperkenankan mengakses *procedure* dengan nama yang diawali “web”.
3. *Procedure* tidak diperkenankan mengakses secara langsung tabel-tabel yang ada dan hanya boleh mengakses *procedure* lain atau *view*.
4. Diperlukan masukan berupa ID desa.

4.3.2.60 Implementasi Procedure web_village_location

Procedure yang dipakai untuk mendapatkan koordinat lokasi dari suatu desa lengkap dengan data waktu tempuh pasukan dari desa pemain tatkala dilakukan penyerangan.

Tabel 4.83 Pseudocode Procedure web_village_location

Pseudocode Procedure web_village_location

Deklarasi awal	
var_idvillage:INT -> Masukan dari user/sistem	
var_target_village:VARCHAR(50) -> Masukan dari user/sistem	
Deskripsi	
1	MULAI
2	
3	SELECT *,
4	get_travel_time(1,var_idvillage,village) AS
5	kris_swordman,
6	get_travel_time(2,var_idvillage,village) AS
7	bhayangkara
8	FROM view_villages_locations
9	WHERE village_name LIKE
10	CONCAT("%",var_target_village,"%")
11	ORDER BY bhayangkara ASC;
12	
13	SELESAI

Penjelasan dari *procedure* web_village_location pada tabel 4.83 yaitu:

1. Merupakan salah satu *procedure* yang boleh diakses langsung oleh aplikasi web.
2. Aplikasi web hanya diperkenankan mengakses *procedure* dengan nama yang diawali “web”.
3. *Procedure* tidak diperkenankan mengakses secara langsung tabel-tabel yang ada dan hanya boleh mengakses *procedure* lain atau view.
4. Diperlukan masukan berupa ID desa dan nama desa yang ingin diketahui lokasinya.

4.3.2.61 Implementasi Procedure web_village_ranking

Procedure yang dipakai untuk mendapatkan peringkat dari desa yang dimiliki oleh pemain.

Tabel 4.84 Pseudocode Procedure web_village_ranking

Pseudocode Procedure web_village_ranking	
Deklarasi awal	
var_idvillage:INT -> Masukan dari user/sistem	
Deskripsi	
1	MULAI
2	
3	CREATE TEMPORARY TABLE temp_villages_rankings (rank
4	INT AUTO_INCREMENT, PRIMARY KEY(rank))
5	AS
6	SELECT *
7	FROM view_villages_rankings;
8	
9	SELECT * FROM temp_villages_rankings WHERE village =
10	var_idvillage;
11	
12	DROP TABLE temp_villages_rankings;
13	
14	SELESAI

Penjelasan dari *procedure* web_village_ranking pada tabel 4.84 yaitu:

1. Merupakan salah satu *procedure* yang boleh diakses langsung oleh aplikasi web.
2. Aplikasi web hanya diperkenankan mengakses *procedure* dengan nama yang diawali “web”.
3. *Procedure* tidak diperkenankan mengakses secara langsung tabel-tabel yang ada dan hanya boleh mengakses *procedure* lain atau view.
4. Diperlukan masukan berupa ID desa.
5. Pembuatan TEMPORARY TABLE pada baris 3 diperlukan untuk membuat urutan angka yang dipakai sebagai peringkat desa.
6. TEMPORARY TABLE yang telah dibuat akan dihapus pada akhir *procedure* di baris 12.

4.3.2.62 Implementasi Procedure web_village_resource_schedules

Procedure yang dipakai untuk menampilkan jadwal dihasilkannya sumber daya tertentu pada desa pemain.

Tabel 4.85 Pseudocode Procedure web_village_resource_schedules

Pseudocode Procedure web_village_resource_schedules	
Deklarasi awal	
var_idvillage:INT -> Masukan dari user/sistem	
var_attribute:VARCHAR(50) -> Masukan dari user/sistem	
var_limit:INT -> Masukan dari user/sistem	
Deskripsi	
1	MULAI
2	
3	SELECT * FROM view_villages_resources_schedules
4	WHERE village = var_idvillage
5	AND attribute = var_attribute
6	ORDER BY execution_time ASC
7	LIMIT var_limit;
8	
9	SELESAI

Penjelasan dari *procedure* web_village_resource_schedules pada tabel 4.85 yaitu:

1. Merupakan salah satu *procedure* yang boleh diakses langsung oleh aplikasi *web*.
2. Aplikasi *web* hanya diperkenankan mengakses *procedure* dengan nama yang diawali “web”.
3. *Procedure* tidak diperkenankan mengakses secara langsung tabel-tabel yang ada dan hanya boleh mengakses *procedure* lain atau *view*.
4. Diperlukan masukan berupa ID desa, jenis sumber daya dan jumlah data yang ingin ditampilkan.

4.3.2.63 Implementasi Procedure web_village_resources

Procedure yang dipakai untuk menampilkan seluruh sumber daya yang dimiliki desa pemain berdasarkan jenisnya.

Tabel 4.86 Pseudocode Procedure web_village_resources

Pseudocode Procedure web_village_resources	
Deklarasi awal	
var_idvillage:INT -> Masukan dari user/sistem	
Deskripsi	
1	MULAI
2	
3	SELECT * FROM view_villages_resources
4	WHERE village = var_idvillage;
5	
6	SELESAI

Penjelasan dari *procedure* web_village_resources pada tabel 4.86 yaitu:

1. Merupakan salah satu *procedure* yang boleh diakses langsung oleh aplikasi *web*.
2. Aplikasi *web* hanya diperkenankan mengakses *procedure* dengan nama yang diawali “web”.
3. *Procedure* tidak diperkenankan mengakses secara langsung tabel-tabel yang ada dan hanya boleh mengakses *procedure* lain atau *view*.
4. Diperlukan masukan berupa ID desa.

4.3.2.64 Implementasi Procedure web_village_troop_build_schedules

Procedure yang dipakai untuk mendapatkan seluruh jadwal pelatihan pasukan pada desa pemain.

Tabel 4.87 Pseudocode Procedure web_village_troop_build_schedules

Pseudocode Procedure web_village_troop_build_schedules	
Deklarasi awal	
var_idvillage:INT -> Masukan dari user/sistem	
Deskripsi	
1	MULAI
2	
3	SELECT *
4	FROM view_villages_troops_builds_schedules
5	WHERE village = var_idvillage
6	AND status = "once";
7	
8	SELESAI

Penjelasan dari *procedure* web_village_troop_build_schedules pada tabel 4.87 yaitu:

1. Merupakan salah satu *procedure* yang boleh diakses langsung oleh aplikasi web.
2. Aplikasi web hanya diperkenankan mengakses *procedure* dengan nama yang diawali "web".
3. *Procedure* tidak diperkenankan mengakses secara langsung tabel-tabel yang ada dan hanya boleh mengakses *procedure* lain atau view.
4. Diperlukan masukan berupa ID desa.

4.3.2.65 Implementasi Procedure web_villages_locations

Procedure yang dipakai untuk mendapatkan koordinat lokasi dari seluruh desa pada *game server online* lengkap dengan data waktu tempuh pasukan dari desa pemain tatkala dilakukan penyerangan.

Tabel 4. 88 Pseudocode Procedure web_villages_locations

Pseudocode Procedure web_villages_locations	
Deklarasi awal	
var_idvillage:INT -> Masukan dari user/sistem	
Deskripsi	
1	MULAI
2	
3	SELECT *,
4	get_travel_time(1,var_idvillage,village) AS
5	kris_swordman,
6	get_travel_time(2,var_idvillage,village) AS
7	bhayangkara
8	FROM view_villages_locations
9	ORDER BY bhayangkara ASC;
10	
11	SELESAI

Penjelasan dari *procedure* web_villages_locations pada tabel 4.88 yaitu:

1. Merupakan salah satu *procedure* yang boleh diakses langsung oleh aplikasi *web*.
2. Aplikasi *web* hanya diperkenankan mengakses *procedure* dengan nama yang diawali “web”.
3. *Procedure* tidak diperkenankan mengakses secara langsung tabel-tabel yang ada dan hanya boleh mengakses *procedure* lain atau *view*.
4. Diperlukan masukan berupa ID desa.

4.3.2.66 Implementasi Procedure web_villages_rankings

Procedure yang dipakai untuk menampilkan data peringkat dari desa yang dimiliki oleh seluruh pemain pada *game server online*.

Tabel 4.89 Pseudocode Procedure web_villages_rankings

Pseudocode Procedure web_villages_rankings	
Deklarasi awal	
Deskripsi	
1	MULAI
2	
3	CREATE TEMPORARY TABLE temp_villages_rankings (rank
4	INT AUTO_INCREMENT, PRIMARY KEY(rank))
5	AS
6	SELECT
7	* FROM view_villages_rankings;
8	
9	SELECT * FROM temp_villages_rankings;
10	
11	DROP TABLE temp_villages_rankings;
12	
13	
14	SELESAI

Penjelasan dari *procedure* web_village_ranking pada tabel 4.89 yaitu:

1. Merupakan salah satu *procedure* yang boleh diakses langsung oleh aplikasi *web*.
2. Aplikasi *web* hanya diperkenankan mengakses *procedure* dengan nama yang diawali “web”.
3. *Procedure* tidak diperkenankan mengakses secara langsung tabel-tabel yang ada dan hanya boleh mengakses *procedure* lain atau *view*.
4. Pembuatan TEMPORARY TABLE pada baris 3 diperlukan untuk membuat urutan angka yang dipakai sebagai peringkat desa.
5. TEMPORARY TABLE yang telah dibuat akan dihapus pada akhir *procedure* di baris 11.

4.3.2.67 Implementasi Procedure web_villages_troops

Procedure yang dipakai untuk menampilkan seluruh jenis pasukan yang dimiliki serta sedang berada di desa pemain lengkap beserta jumlahnya.

Tabel 4.90 Pseudocode Procedure web_villages_troops

Pseudocode Procedure web_villages_troops	
Deklarasi awal	
var_idvillage:INT -> Masukan dari user/sistem	
Deskripsi	
1	MULAI
2	
3	SELECT * FROM view_villages_troops WHERE village =
4	var_idvillage;
5	
6	SELESAI

Penjelasan dari *procedure* web_villages_troops pada tabel 4.90 yaitu:

1. Merupakan salah satu *procedure* yang boleh diakses langsung oleh aplikasi *web*.
2. Aplikasi *web* hanya diperkenankan mengakses *procedure* dengan nama yang diawali “web”.
3. *Procedure* tidak diperkenankan mengakses secara langsung tabel-tabel yang ada dan hanya boleh mengakses *procedure* lain atau *view*.
4. Diperlukan masukan berupa ID desa.

4.3.2.68 Implementasi Procedure web_war_loot_schedules

Procedure yang dipakai untuk menampilkan jadwal kedatangan hasil sumber daya rampasan perang di desa pemain.

Tabel 4.91 Pseudocode Procedure web_war_loot_schedules

Pseudocode Procedure web_war_loot_schedules	
Deklarasi awal	
var_idvillage:INT -> Masukan dari user/sistem	
Deskripsi	
1	MULAI
2	
3	SELECT *
4	FROM view_wars_loots_schedules
5	WHERE village = var_idvillage AND
6	STATUS = "once";
7	
8	SELESAI

Penjelasan dari *procedure* web_war_loot_schedules pada tabel 4.91 yaitu:

1. Merupakan salah satu *procedure* yang boleh diakses langsung oleh aplikasi web.
2. Aplikasi web hanya diperkenankan mengakses *procedure* dengan nama yang diawali “web”.
3. *Procedure* tidak diperkenankan mengakses secara langsung tabel-tabel yang ada dan hanya boleh mengakses *procedure* lain atau view.
4. Diperlukan masukan berupa ID desa.

4.3.3 Event

Ragam algoritma *event* dalam *game server online* permainan *strategy* Kerajaan Majapahit ialah sebagai berikut.

4.3.3.1 Implementasi Event event_sync_all_mem_variables

Event yang dipakai untuk melakukan pemanggilan *procedure* sync_all_mem_variables secara berkala.

Tabel 4.92 Pseudocode Event event_sync_all_mem_variables

Pseudocode Event event_sync_all_mem_variables	
Deskripsi	
1	MULAI
2	
3	CALL sync_all_mem_variables()
4	
5	SELESAI

Penjelasan dari *event* event_sync_all_mem_variables pada tabel 4.92 yaitu:

1. Merupakan EVENT yang memiliki tipe RECURRING.
2. EVENT di eksekusi tiap 5 menit sekali.

4.3.3.2 Implementasi Event event_villages_buildings_builds_scheduling

Event yang dipakai untuk melakukan pemanggilan *procedure* villages_buildings_builds_scheduling secara berkala.

Tabel 4.93 Pseudocode Event event_villages_buildings_builds_scheduling

Pseudocode Event event_villages_buildings_builds_scheduling	
Deskripsi	
1	MULAI
2	
3	CALL villages_buildings_builds_scheduling()
4	
5	SELESAI

Penjelasan dari *event* event_villages_buildings_builds_scheduling pada tabel 4.93 yaitu:

1. Merupakan EVENT yang memiliki tipe RECURRING.
2. EVENT di eksekusi tiap detik.

4.3.3.3 Implementasi Event event_villages_resources_scheduling

Event yang dipakai untuk melakukan pemanggilan *procedure* villages_resources_scheduling secara berkala.

Tabel 4.94 Pseudocode Event event_villages_resources_scheduling

Pseudocode Event event_villages_resources_scheduling	
Deskripsi	
1	MULAI
2	
3	CALL villages_resources_scheduling()
4	
5	SELESAI

Penjelasan dari *event* event_villages_resources_scheduling pada tabel 4.94 yaitu:

1. Merupakan EVENT yang memiliki tipe RECURRING.
2. EVENT di eksekusi tiap detik.

4.3.3.4 Implementasi Event event_villages_troops_attacks_scheduling

Event yang dipakai untuk melakukan pemanggilan *procedure* villages_troops_attacks_scheduling secara berkala.

Tabel 4.95 Pseudocode Event event_villages_troops_attacks_scheduling

Pseudocode Event event_villages_troops_attacks_scheduling	
Deskripsi	
1	MULAI
2	
3	CALL villages_troops_attacks_scheduling()
4	
5	SELESAI

Penjelasan dari *event* `event_villages_troops_attacks_scheduling` pada tabel 4.95 yaitu:

1. Merupakan EVENT yang memiliki tipe RECURRING.
2. EVENT di eksekusi tiap detik.

4.3.3.5 Implementasi Event `event_villages_troops_builds_scheduling`

Event yang dipakai untuk melakukan pemanggilan *procedure* `villages_troops_builds_scheduling` secara berkala.

Tabel 4.96 Pseudocode Event `event_villages_troops_builds_scheduling`

Pseudocode Event <code>event_villages_troops_builds_scheduling</code>	
Deskripsi	
1	MULAI
2	
3	CALL <code>villages_troops_builds_scheduling()</code>
4	
5	SELESAI

Penjelasan dari *event* `event_villages_troops_builds_scheduling` pada tabel 4.96 yaitu:

1. Merupakan EVENT yang memiliki tipe RECURRING.
2. EVENT di eksekusi tiap detik.

4.3.3.6 Implementasi Event `event_wars_loots_scheduling`

Event yang dipakai untuk melakukan pemanggilan *procedure* `wars_loots_scheduling` secara berkala.

Tabel 4.97 Pseudocode Event event_wars_loots_scheduling

Pseudocode Event event_wars_loots_scheduling	
Deskripsi	
1	MULAI
2	
3	CALL wars_loots_scheduling()
4	
5	SELESAI

Penjelasan dari *event* event_wars_loots_scheduling pada tabel 4.97 yaitu:

1. Merupakan EVENT yang memiliki tipe RECURRING.
2. EVENT di eksekusi tiap detik.

4.4 Implementasi Web

Implementasi *web* yang dikembangkan dengan MVC *framework* Code Igniter terdiri dari berbagai ragam algoritma kelas *model*, kelas *controller* dan *view*. Dalam subbab ini akan dijelaskan algoritma dari keseluruhan kelas *model*, kelas *controller* serta *view* yang ada dalam aplikasi *web game server online* permainan *strategy* Kerajaan Majapahit. Adapun ragam algoritma tersebut ialah:

4.4.1 Implementasi Kelas Model

Ragam algoritma kelas *model* dalam aplikasi *web game server online* permainan *strategy* Kerajaan Majapahit ialah sebagai berikut.

4.4.1.1 Implementasi Kelas Buildings_model

Merupakan kelas yang berhubungan langsung dengan berbagai *procedure database* yang terkait dengan objek bangunan pada permainan.

Tabel 4. 98 Pseudocode Class Buildings_model

Pseudocode Class Buildings_model

	Deskripsi
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45	MULAI class Buildings_model extends CI_model { function __construct(){ parent::__construct(); \$this->load->database(); } function village_building_amounts(\$idvillage,\$building){ \$query = \$this->db->query("CALL web_village_building_amounts ('{\$idvillage}','{\$building}')"); return \$query->result(); } function building_prices(){ \$query = \$this->db->query("CALL web_building_prices()"); return \$query->result(); } function village_building_build_schedules(\$idvillage){ \$query = \$this->db->query("CALL web_village_building_build_schedules ('{\$idvillage}')"); return \$query->result(); } function count_village_building_build_schedules(\$idvillage){ \$query = \$this->db->query("CALL web_count_village_building_build_schedules ('{\$idvillage}')");

```

46
47         return $query->result();
48     }
49
50     function order_building($idvillage,$building) {
51
52         $query = $this->db->query("CALL
53         web_order_building
54         ('$idvillage', '$building')");
55
56         return $query->result();
57     }
58 }
59
60 SELESAI

```

Penjelasan dari kelas Buildings_model pada tabel 4.98 yaitu:

1. Merupakan turunan dari kelas CI_model.
2. Berisi berbagai *method* terkait objek bangunan pada permainan yang nantinya digunakan pada kelas *controller*.
3. Kinerja semua *method* secara garis besar ialah melakukan pemanggilan *query* eksekusi *procedure*.
4. Bila pemanggilan *query* menghasilkan nilai, nilai tersebut akan menjadi nilai kembalian *method*.

4.4.1.2 Implementasi Kelas Form_validation_model

Merupakan kelas yang berhubungan langsung dengan berbagai *procedure database* yang terkait dengan sistem validasi pada permainan.

Tabel 4. 99 Pseudocode Class Form_validation_model

Pseudocode Class Form_validation_model	
Deskripsi	
1	MULAI
2	


```

3  class Form_validation_model extends CI_Model {
4
5      function __construct(){
6
7          parent::__construct();
8
9          $this->load->database();
10     }
11
12     function username_availability($username){
13
14         $query = $this->db->query("CALL
15         web_username_availability
16         ('$username',@availability)");
17
18         $query = $this->db->query("SELECT
19         @availability AS availability");
20
21         return $query->result();
22     }
23 }
24
25 SELESAI

```

Penjelasan dari kelas Form_validation_model pada tabel 4.99 yaitu:

1. Merupakan turunan dari kelas CI_model.
2. Berisi berbagai *method* terkait dengan sistem validasi pada permainan yang nantinya digunakan pada kelas *controller*.
3. Kinerja semua *method* secara garis besar ialah melakukan pemanggilan *query* eksekusi *procedure*.
4. Bila pemanggilan *query* menghasilkan nilai, nilai tersebut akan menjadi nilai kembalian *method*.
5. *Query* SELECT pada baris 18-19 berfungsi untuk mendapatkan nilai keluaran dari *procedure*.

4.4.1.3 Implementasi Kelas Log_in_model

Merupakan kelas yang berhubungan langsung dengan berbagai *procedure database* yang terkait dengan sistem identifikasi pemain pada permainan.

Tabel 4. 100 Pseudocode Class Log_in_model

Pseudocode Class Log_in_model	
Deskripsi	
1	MULAI
2	
3	class Log_in_model extends CI_model {
4	function __construct(){
5	parent::__construct();
6	\$this->load->database();
7	}
8	function log_in(\$username,\$password){
9	\$query = \$this->db->query("CALL
10	web_log_in
11	('\$username','\$password',@status)");
12	\$query = \$this->db->query
13	("SELECT @status AS status");
14	return \$query->result();
15	}
16	}
17	
18	SELESAI

Penjelasan dari kelas Log_in_model pada tabel 4.100 yaitu:

1. Merupakan turunan dari kelas CI_model.

2. Berisi berbagai *method* terkait dengan sistem identifikasi pemain pada permainan yang nantinya digunakan pada kelas *controller*.
3. Kinerja semua *method* secara garis besar ialah melakukan pemanggilan *query* eksekusi *procedure*.
4. Bila pemanggilan *query* menghasilkan nilai, nilai tersebut akan menjadi nilai kembalian *method*.
5. *Query* SELECT pada baris 18-19 berfungsi untuk mendapatkan nilai keluaran dari *procedure*.

4.4.1.4 Implementasi Kelas *Resources_model*

Merupakan kelas yang berhubungan langsung dengan berbagai *procedure database* yang terkait dengan objek sumber daya pada permainan.

Tabel 4.101 Pseudocode Class *Resources_model*

Pseudocode Class <i>Resources_model</i>	
Deskripsi	
1	MULAI
2	
3	<code>class Resources_model extends CI_model {</code>
4	<code>function __construct(){</code>
5	<code>parent::__construct();</code>
6	<code>\$this->load->database();</code>
7	<code>}</code>
8	<code>function village_resources(\$idvillage){</code>
9	<code>\$query = \$this->db->query("CALL</code>
10	<code>web_village_resources('\$idvillage');</code>
11	<code>return \$query->result();</code>
12	<code>}</code>
13	<code>function village resource schedules</code>
14	
15	
16	
17	
18	
19	
20	


```

21      ($idvillage,$resource_type,$limit){
22
23          $query = $this->db->query("CALL
24              web_village_resource_schedules
25              ('$idvillage','$resource_type','$limit')");
26
27          return $query->result();
28      }
29
30      function war_loot_schedules($idvillage){
31
32          $query = $this->db->query("CALL
33              web_war_loot_schedules('$idvillage')");
34
35          return $query->result();
36      }
37
38      function count_war_loot_schedules($idvillage){
39
40          $query = $this->db->query("CALL
41              web_count_war_loot_schedules
42              ('$idvillage')");
43
44          return $query->result();
45      }
46
47  }
48
49  SELESAI

```

Penjelasan dari kelas Resources_model pada tabel 4.101 yaitu:

1. Merupakan turunan dari kelas CI_model.
2. Berisi berbagai *method* terkait dengan objek sumber daya pada permainan yang nantinya digunakan pada kelas *controller*.
3. Kinerja semua *method* secara garis besar ialah melakukan pemanggilan *query* eksekusi *procedure*.
4. Bila pemanggilan *query* menghasilkan nilai, nilai tersebut akan menjadi nilai kembalian *method*.

4.4.1.5 Implementasi Kelas Sign_up_model

Merupakan kelas yang berhubungan langsung dengan berbagai *procedure database* yang terkait dengan pendaftaran calon permainan di *game server online*.

Tabel 4.102 Pseudocode Class Sign_up_model

Pseudocode Class Sign_up_model	
Deskripsi	
1	MULAI
2	
3	class Sign_up_model extends CI_model {
4	function __construct(){
5	parent::__construct();
6	\$this->load->database();
7	}
8	
9	Function
10	sign_up(\$username,\$password,\$password_conf){
11	
12	\$query = \$this->db->query("CALL
13	web_sign_up('\$username','\$password',
14	'\$password_conf',@status)");
15	
16	\$query = \$this->db->query
17	("SELECT @status AS status");
18	
19	return \$query->result();
20	
21	}
22	
23	}
24	
25	
26	
27	SELESAI

Penjelasan dari kelas Sign_up_model pada tabel 4.102 yaitu:

1. Merupakan turunan dari kelas CI_model.

2. Berisi berbagai *method* terkait dengan pendaftaran calon pemain pada permainan yang nantinya digunakan pada kelas *controller*.
3. Kinerja semua *method* secara garis besar ialah melakukan pemanggilan *query* eksekusi *procedure*.
4. Bila pemanggilan *query* menghasilkan nilai, nilai tersebut akan menjadi nilai kembalian *method*.
5. *Query* SELECT pada baris 19-20 berfungsi untuk mendapatkan nilai keluaran dari *procedure*.

4.4.1.6 Implementasi Kelas Troops_model

Merupakan kelas yang berhubungan langsung dengan berbagai *procedure database* yang terkait dengan objek pasukan pada permainan.

Tabel 4.103 Pseudocode Class Troops_model

Pseudocode Class Troops_model	
Deskripsi	
1	MULAI
2	
3	class Troops_model extends CI_model {
4	function __construct() {
5	parent::__construct();
6	\$this->load->database();
7	}
8	function troop_attack_schedules(\$idvillage) {
9	\$query = \$this->db->query("CALL
10	web_troop_attack_schedules('\$idvillage')");
11	return \$query->result();
12	}
13	
14	function
15	
16	
17	
18	
19	
20	


```

21     count_troop_attack_schedules($idvillage){
22
23         $query = $this->db->query("CALL
24         web_count_troop_attack_schedules
25         ('$idvillage')");
26
27         return $query->result();
28     }
29
30     function enemy_attack_schedules($idvillage){
31
32         $query = $this->db->query("CALL
33         web_enemy_attack_schedules('$idvillage')");
34
35         return $query->result();
36     }
37
38     function villages_troops($idvillage){
39
40         $query = $this->db->query("CALL
41         web_villages_troops('$idvillage')");
42
43         return $query->result();
44     }
45
46     function
47     count_enemy_attack_schedules($idvillage){
48
49         $query = $this->db->query("CALL
50         web_count_enemy_attack_schedules
51         ('$idvillage')");
52
53         return $query->result();
54     }
55
56     function troop_back_schedules($idvillage){
57
58         $query = $this->db->query("CALL
59         web_troop_back_schedules('$idvillage')");
60
61         return $query->result();
62     }
63
64     function count_troop_back_schedules($idvillage){
65
66         $query = $this->db->query("CALL
67         web count troop back schedules

```

```
68         ('$idvillage'))";
69
70         return $query->result();
71     }
72
73     function troop_prices(){
74
75         $query = $this->db->query("CALL
76         web_troop_prices()");
77
78         return $query->result();
79     }
80
81     function troop_stats(){
82
83         $query = $this->db->query("CALL
84         web_troop_stats()");
85
86         return $query->result();
87     }
88
89     function
90     village_troop_build_schedules($idvillage){
91
92         $query = $this->db->query("CALL
93         web_village_troop_build_schedules
94         ('$idvillage')");
95
96         return $query->result();
97     }
98
99     function
100     count_village_troop_build_schedules($idvillage){
101
102         $query = $this->db->query("CALL
103         web_count_village_troop_build_schedules
104         ('$idvillage')");
105
106         return $query->result();
107     }
108
109     function order_troop($idvillage,$stroop){
110
111         $query = $this->db->query("CALL
112         web_order_troop('$idvillage','$stroop')");
113
114         return $query->result();
```

115	}
116	function
117	order_attack(\$idvillage,\$troop,\$target_village){
118	
119	\$query = \$this->db->query("CALL
120	web_order_attack
121	('\$idvillage','\$troop','\$target_
122	village')");
123	
124	return \$query->result();
125	}
126	}
127	SELESAI

Penjelasan dari kelas Troops_model pada tabel 4.103 yaitu:

1. Merupakan turunan dari kelas CI_model.
2. Berisi berbagai *method* terkait dengan objek pasukan pada permainan yang nantinya digunakan pada kelas *controller*.
3. Kinerja semua *method* secara garis besar ialah melakukan pemanggilan *query* eksekusi *procedure*.
4. Bila pemanggilan *query* menghasilkan nilai, nilai tersebut akan menjadi nilai kembalian *method*.

4.4.1.7 Implementasi Kelas Villages_model

Merupakan kelas yang berhubungan langsung dengan berbagai *procedure database* yang terkait dengan objek desa pemain.

Tabel 4.104 Pseudocode Class Villages_model

Pseudocode Class Villages_model	
Deskripsi	
1	MULAI
2	
3	class Villages_model extends CI_model {
4	


```
5      function __construct() {
6
7          parent::__construct();
8
9          $this->load->database();
10     }
11
12     function default_active_village($username) {
13
14         $query = $this->db->query("CALL
15         web_default_active_village
16         ('$username', @idvillage)");
17
18         $query = $this->db->query("SELECT @idvillage
19         AS idvillage");
20
21         return $query->result();
22     }
23
24     function village_ranking($idvillage) {
25
26         $query = $this->db->query("CALL
27         web_village_ranking('$idvillage')");
28
29         return $query->result();
30     }
31
32     function villages_rankings() {
33
34         $query = $this->db->query("CALL
35         web_villages_rankings()");
36
37         return $query->result();
38     }
39
40     function villages_locations($idvillage) {
41
42         $query = $this->db->query("CALL
43         web_villages_locations('$idvillage')");
44
45         return $query->result();
46     }
47
48     function
49     village_location($idvillage, $targetvillage) {
50
51         $query = $this->db->query("CALL
```

```

52      web_village_location
53      ('$idvillage', '$targetvillage'));
54
55      return $query->result();
56  }
57
58  }
59
60  SELESAI

```

Penjelasan dari kelas *Villages_model* pada tabel 4.104 yaitu:

1. Merupakan turunan dari kelas *CI_model*.
2. Berisi berbagai *method* terkait dengan objek desa pada permainan yang nantinya digunakan pada kelas *controller*.
3. Kinerja semua *method* secara garis besar ialah melakukan pemanggilan *query* eksekusi *procedure*.
4. Bila pemanggilan *query* menghasilkan nilai, nilai tersebut akan menjadi nilai kembalian *method*.
5. *Query* *SELECT* pada baris 18-19 berfungsi untuk mendapatkan nilai keluaran dari *procedure*.

4.4.2 Implementasi Kelas *Controller*

Ragam algoritma kelas *controller* dalam aplikasi *web game server online* permainan *strategy* Kerajaan Majapahit ialah sebagai berikut.

4.4.2.1 Implementasi Kelas *Buildings*

Merupakan kelas penampung berbagai *function* terkait objek bangunan yang berhubungan langsung serta menjembatani antara kelas *Buildings_model* dengan berbagai *compatible game client*.

Tabel 4.105 Pseudocode Class *Buildings*

Pseudocode Class <i>Buildings</i>

Deskripsi	
1	MULAI
2	
3	class Buildings extends CI_controller{
4	public function __construct(){
5	parent::__construct();
6	\$this->load->model("buildings_model");
7	}
8	public function index(){
9	redirect(base_url());
10	}
11	public function
12	player_village_building_amounts(){
13	\$idvillage = \$this->input->
14	post("idvillage");
15	\$building = \$this->input->post("building");
16	\$building_amounts = \$this->buildings_model->
17	village_building_amounts
18	(\$idvillage,\$building);
19	\$this->output->set_content_type
20	("application/json")->
21	set_output(json_encode(\$building_amounts));
22	}
23	public function player_building_prices(){
24	\$building_prices = \$this->buildings_model->
25	building_prices();
26	\$this->output->set_content_type
27	("application/json")
28	->set_output(json_encode(\$building_prices));
29	}
30	public function player_village_building


```

46     build_schedules() {
47
48         $idvillage = $this->input->
49         post("idvillage");
50
51         $village_building_build_schedules = $this->
52         buildings_model->
53         village_building_build_schedules
54         ($idvillage);
55
56         $this->output->set_content_type
57         ("application/json")->set_output(json_encode
58         ($village_building_build_schedules));
59     }
60
61     public function
62     player_count_village_building_build_schedules() {
63
64         $idvillage = $this->input->
65         post("idvillage");
66
67         $count_village_building_build_schedules =
68         $this->buildings_model->
69         count_village_building_build_schedules
70         ($idvillage);
71
72         $this->output->set_content_type
73         ("application/json")->
74         set_output(json_encode
75         ($count_village_building_build_schedules));
76     }
77
78     public function player_order_building() {
79         $idvillage = $this->input->
80         post("idvillage");
81
82         $building = $this->input->post("building");
83         $order_building = $this->buildings_model->
84         order_building($idvillage,$building);
85     }
86 }
87
88 SELESAI

```

Penjelasan dari kelas Buildings pada tabel 4.105 yaitu:

1. Merupakan turunan dari kelas *CI_controller*.
2. Berisi berbagai *method* terkait objek bangunan pada permainan yang nantinya bisa diakses oleh *game client*.
3. Kinerja semua *method* secara garis besar ialah menerima masukan dari *game client* untuk diteruskan ke *function* yang ada pada kelas *Buildings_model*.
4. Masukan diterima *function* dari metode POST dengan menggunakan *function* post kelas Input Code Igniter.
5. Bilamana ada nilai keluaran untuk *game client*, *function* akan memberi keluaran dalam format JSON menggunakan kelas Output Code Igniter.
6. Untuk mengubah nilai balik dari *function-function* kelas *Buildings_model* menjadi format JSON digunakan *function* *json_encode*.

4.4.2.2 Implementasi Kelas *Form_validation*

Merupakan kelas penampung berbagai *function* terkait sistem validasi permainan yang berhubungan langsung serta menjembatani antara kelas *Form_validation_model* dengan berbagai *compatible game client*.

Tabel 4.106 Pseudocode Class *Form_validation*

Pseudocode Class <i>Form_validation</i>	
Deskripsi	
1	MULAI
2	
3	<code>class Form_validation extends CI_Controller {</code>
4	
5	
6	<code> public function __construct(){</code>
7	
8	<code> parent::__construct();</code>
9	
10	<code> \$this->load->model("form_validation_model");</code>
11	<code> }</code>
12	
13	<code> public function index(){</code>
14	

```

15         redirect(base_url());
16     }
17
18     public function username_availability(){
19
20         $username = $this->input->post("username");
21
22         $availability = $this->
23         form_validation_model->
24         username_availability($username);
25
26         $this->output->
27         set_content_type("application/json")->
28         set_output(json_encode($availability));
29
30     }
31 }
32 SELESAI

```

Penjelasan dari kelas Form_validation pada tabel 4.106 yaitu:

1. Merupakan turunan dari kelas CI_controller.
2. Berisi berbagai *method* terkait sistem validasi pada permainan yang nantinya bisa diakses oleh *game client*.
3. Kinerja semua *method* secara garis besar ialah menerima masukan dari *game client* untuk diteruskan ke *function* yang ada pada kelas Form_validation_model.
4. Masukan diterima *function* dari metode POST dengan menggunakan *function* post kelas Input Code Igniter.
5. Bilamana ada nilai keluaran untuk *game client*, *function* akan memberi keluaran dalam format JSON menggunakan kelas Output Code Igniter.
6. Untuk mengubah nilai balik dari *function-function* kelas Form_validation_model menjadi format JSON digunakan *function* json_encode.

4.4.2.3 Implementasi Kelas Home

Merupakan kelas yang berfungsi untuk menampilkan view yang berisi pedoman penggunaan kelas dan *method* yang ada pada *game server online* oleh *game client*.

Tabel 4.107 Pseudocode Class Home

Pseudocode Class Home	
Deskripsi	
1	MULAI
2	
3	class Home extends CI_Controller {
4	public function __construct(){
5	parent::__construct();
6	}
7	public function index(){
8	\$this->load->view('home');
9	}
10	}
11	
12	SELESAI
13	
14	
15	
16	
17	

Penjelasan dari kelas Home pada tabel 4.107 yaitu:

1. Merupakan turunan dari kelas CI_controller.
2. Kinerja kelas secara garis besar ialah menerima pesanan *game client* untuk menampilkan view.

4.4.2.4 Implementasi Kelas Log_in

Merupakan kelas penampung berbagai *function* terkait sistem identifikasi pemain yang berhubungan langsung serta menjembatani antara kelas Log_in_model dengan berbagai *compatible game client*.

Tabel 4.108 Pseudocode Class Log_in

Pseudocode Class Log_in	
Deskripsi	
1	MULAI
2	
3	class Log_in extends CI_controller{
4	public function __construct(){
5	parent::__construct();
6	\$this->load->model("log_in_model");
7	}
8	public function index(){
9	redirect(base_url());
10	}
11	public function player_log_in(){
12	\$username = \$this->input->post("username");
13	\$password = \$this->input->post("password");
14	\$status = \$this->log_in_model->
15	log_in(\$username,\$password);
16	\$this->output->
17	set_content_type("application/json")->
18	set_output(json_encode(\$status));
19	}
20	}
21	
22	
23	
24	
25	
26	
27	
28	
29	
30	
31	
32	
33	

34	SELESAI

Penjelasan dari kelas Log_in pada tabel 4.108 yaitu:

1. Merupakan turunan dari kelas CI_controller.
2. Berisi berbagai *method* terkait sistem identifikasi pemain yang nantinya bisa diakses oleh *game client*.
3. Kinerja semua *method* secara garis besar ialah menerima masukan dari *game client* untuk diteruskan ke *function* yang ada pada kelas Log_in_model.
4. Masukan diterima *function* dari metode POST dengan menggunakan *function* post kelas Input Code Igniter.
5. Bilamana ada nilai keluaran untuk *game client*, *function* akan memberi keluaran dalam format JSON menggunakan kelas Output Code Igniter.
6. Untuk mengubah nilai balik dari *function-function* kelas Log_in_model menjadi format JSON digunakan *function* json_encode.

4.4.2.5 Implementasi Kelas Resources

Merupakan kelas penampung berbagai *function* terkait objek sumber daya pada permainan yang berhubungan langsung serta menjembatani antara kelas Resources_model dengan berbagai *compatible game client*.

Tabel 4.109 Pseudocode Class Resources

Pseudocode Class Resources	
Deskripsi	
1	MULAI
2	
3	class Resources extends CI_controller{
4	public function __construct(){
5	parent::__construct();
6	}
7	
8	


```
9         $this->load->model("resources_model");
10     }
11
12     public function index(){
13
14         redirect(base_url());
15     }
16
17     public function player_village_resources(){
18
19         $idvillage = $this->input->
20         post("idvillage");
21
22         $resources = $this->resources_model->
23         village_resources($idvillage);
24
25         $this->output->set_content_type
26         ("application/json")->
27         set_output(json_encode($resources));
28     }
29
30
31     public function
32     player_village_resource_schedules(){
33
34         $idvillage=$this->input->post("idvillage");
35
36         $resource_type=$this->input->
37         post("resource_type");
38
39         $limit=$this->input->post("limit");
40
41         $schedules = $this->resources_model->
42         village_resource_schedules
43         ($idvillage,$resource_type,$limit);
44
45         $this->output->
46         set_content_type("application/json")->
47         set_output(json_encode($schedules));
48     }
49
50     public function player_war_loot_schedules(){
51
52         $idvillage=$this->input->post("idvillage");
53
54         $schedules = $this->resources_model->
55         war_loot_schedules($idvillage);
```

```

56
57     $this->output->
58     set_content_type("application/json")->
59     set_output(json_encode($schedules));
60 }
61
62 public function
63 player_count_war_loot_schedules() {
64
65     $idvillage=$this->input->post("idvillage");
66
67     $schedules = $this->resources_model->
68     count_war_loot_schedules($idvillage);
69
70     $this->output->
71     set_content_type("application/json")->
72     set_output(json_encode($schedules));
73 }
74
75 }
76
77 SELESAI

```

Penjelasan dari kelas Resources pada tabel 4.109 yaitu:

1. Merupakan turunan dari kelas CI_controller.
2. Berisi berbagai *method* terkait sumber daya permainan yang nantinya bisa diakses oleh *game client*.
3. Kinerja semua *method* secara garis besar ialah menerima masukan dari *game client* untuk diteruskan ke *function* yang ada pada kelas Resources_model.
4. Masukan diterima *function* dari metode POST dengan menggunakan *function* post kelas Input Code Igniter.
5. Bilamana ada nilai keluaran untuk *game client*, *function* akan memberi keluaran dalam format JSON menggunakan kelas Output Code Igniter.
6. Untuk mengubah nilai balik dari *function-function* kelas Resources_model menjadi format JSON digunakan *function* json_encode.

4.4.2.6 Implementasi Kelas Sign_up

Merupakan kelas penampung berbagai *function* terkait sistem pendaftaran calon pemain yang berhubungan langsung serta menjembatani antara kelas Sign_up_model dengan berbagai *compatible game client*.

Tabel 4.110 Pseudocode Class Sign_up

Pseudocode Class Sign_up	
Deskripsi	
1	MULAI
2	
3	class Sign_up extends CI_controller{
4	public function __construct(){
5	parent::__construct();
6	\$this->load->model("sign_up_model");
7	}
8	public function index(){
9	redirect(base_url());
10	}
11	public function player_sign_up(){
12	\$username = \$this->input->post("username");
13	\$password = \$this->input->post("password");
14	\$password_conf = \$this->input->
15	post("password_conf");
16	\$status = \$this->sign_up_model->
17	sign_up(\$username,\$password,\$password_conf);
18	\$this->output->
19	set_content_type("application/json")->
20	set_output(json_encode(\$status));
21	}
22	
23	
24	
25	
26	
27	
28	
29	
30	
31	
32	
33	

34	
35	}
36	
37	SELESAI

Penjelasan dari kelas Sign_up pada tabel 4.110 yaitu:

1. Merupakan turunan dari kelas CI_controller.
2. Berisi berbagai *method* terkait sistem pendaftaran pemain yang nantinya bisa diakses oleh *game client*.
3. Kinerja semua *method* secara garis besar ialah menerima masukan dari *game client* untuk diteruskan ke *function* yang ada pada kelas Sign_up_model.
4. Masukan diterima *function* dari metode POST dengan menggunakan *function* post kelas Input Code Igniter.
5. Bilamana ada nilai keluaran untuk *game client*, *function* akan memberi keluaran dalam format JSON menggunakan kelas Output Code Igniter.
6. Untuk mengubah nilai balik dari *function-function* kelas Sign_up_model menjadi format JSON digunakan *function* json_encode.

4.4.2.7 Implementasi Kelas Troops

Merupakan kelas penampung berbagai *function* terkait objek pasukan pada permainan yang berhubungan langsung serta menjembatani antara kelas Troops_model dengan berbagai *compatible game client*.

Tabel 4.111 Pseudocode Class Troops

Pseudocode Class Troops	
Deskripsi	
1	MULAI
2	
3	class Troops extends CI_controller{
4	
5	public function construct(){

```

6
7         parent::__construct();
8
9         $this->load->model("troops_model");
10    }
11
12    public function index(){
13
14        redirect(base_url());
15    }
16
17    public function
18    player_troop_attack_schedules(){
19
20        $idvillage=$this->input->
21        post("idvillage");
22
23        $schedules = $this->troops_model->
24        troop_attack_schedules($idvillage);
25
26        $this->output->
27        set_content_type("application/json")->
28        set_output(json_encode($schedules));
29    }
30
31    public function
32    player_count_troop_attack_schedules(){
33
34        $idvillage=$this->input->
35        post("idvillage");
36
37        $schedules = $this->troops_model->
38        count_troop_attack_schedules($idvillage);
39
40        $this->output->
41        set_content_type("application/json")->
42        set_output(json_encode($schedules));
43    }
44
45    public function
46    player_enemy_attack_schedules(){
47
48        $idvillage=$this->input->
49        post("idvillage");
50
51        $schedules = $this->troops_model->
52        enemy_attack_schedules($idvillage);

```

```

53
54         $this->output->
55         set_content_type("application/json")->
56         set_output(json_encode($schedules));
57     }
58
59     public function
60     player_count_enemy_attack_schedules() {
61
62         $idvillage=$this->input->
63         post("idvillage");
64
65         $schedules = $this->troops_model->
66         count_enemy_attack_schedules($idvillage);
67
68         $this->output->
69         set_content_type("application/json")->
70         set_output(json_encode($schedules));
71     }
72
73     public function player_troop_back_schedules(){
74
75         $idvillage=$this->input->
76         post("idvillage");
77
78         $schedules = $this->troops_model->
79         troop_back_schedules($idvillage);
80
81         $this->output->
82         set_content_type("application/json")->
83         set_output(json_encode($schedules));
84     }
85
86     public function
87     player_count_troop_back_schedules() {
88
89         $idvillage=$this->input->
90         post("idvillage");
91
92         $schedules = $this->troops_model->
93         count_troop_back_schedules($idvillage);
94
95         $this->output->
96         set_content_type("application/json")->
97         set_output(json_encode($schedules));
98     }
99

```



```
100     public function player_troop_prices() {
101
102         $troop_prices = $this->troops_model->
103         troop_prices();
104
105         $this->output->
106         set_content_type("application/json")->
107         set_output(json_encode($troop_prices));
108     }
109
110     public function player_troop_stats() {
111
112         $troop_prices = $this->troops_model->
113         troop_stats();
114
115         $this->output->
116         set_content_type("application/json")->
117         set_output(json_encode($troop_prices));
118     }
119
120     public function
121     player_village_troop_build_schedules() {
122
123         $idvillage=$this->input->
124         post("idvillage");
125
126         $schedules = $this->troops_model->
127         village_troop_build_schedules($idvillage);
128
129         $this->output->
130         set_content_type("application/json")->
131         set_output(json_encode($schedules));
132     }
133
134     public function
135     player_count_village_troop_build_schedules() {
136
137         $idvillage=$this->input->
138         post("idvillage");
139
140         $schedules = $this->troops_model->
141         count_village_troop_build_schedules
142         ($idvillage);
143
144         $this->output->
145         set_content_type("application/json")->
146         set_output(json_encode($schedules));
```

```
147     }
148
149     public function player_order_troop(){
150
151         $idvillage=$this->input->
152         post("idvillage");
153
154         $troop=$this->input->post("troop");
155
156         $order = $this->troops_model->
157         order_troop($idvillage,$troop);
158
159         $this->output->
160         set_content_type("application/json")->
161         set_output(json_encode($order));
162     }
163
164     public function player_order_attack(){
165
166         $idvillage=$this->input->
167         post("idvillage");
168
169         $troop=$this->input->post("troop");
170
171         $target_village=$this->input->
172         post("target_village");
173
174         $order = $this->troops_model->
175         order_attack
176         ($idvillage,$troop,$target_village);
177
178         $this->output->
179         set_content_type("application/json")->
180         set_output(json_encode($order));
181     }
182
183     public function player_villages_troops(){
184
185         $idvillage=$this->input->
186         post("idvillage");
187
188         $villages_troops = $this->troops_model->
189         villages_troops($idvillage);
190
191         $this->output->
192         set_content_type("application/json")->
193         set_output(json_encode($villages_troops));
```

194	}
195	
196	}
197	
198	SELESAI

Penjelasan dari kelas Troops pada tabel 4.111 yaitu:

1. Merupakan turunan dari kelas CI_controller.
2. Berisi berbagai *method* terkait objek pasukan pada permainan yang nantinya bisa diakses oleh *game client*.
3. Kinerja semua *method* secara garis besar ialah menerima masukan dari *game client* untuk diteruskan ke *function* yang ada pada kelas Troops_model.
4. Masukan diterima *function* dari metode POST dengan menggunakan *function* post kelas Input Code Igniter.
5. Bilamana ada nilai keluaran untuk *game client*, *function* akan memberi keluaran dalam format JSON menggunakan kelas Output Code Igniter.
6. Untuk mengubah nilai balik dari *function-function* kelas Troops_model menjadi format JSON digunakan *function* json_encode.

4.4.2.8 Implementasi Kelas Villages

Merupakan kelas penampung berbagai *function* terkait objek desa pemain yang berhubungan langsung serta menjembatani antara kelas Villages_model dengan berbagai *compatible game client*.

Tabel 4.112 Pseudocode Class Villages

Pseudocode Class Villages	
Deskripsi	
1	MULAI
2	
3	class Villages extends CI_controller{
4	


```
5      public function __construct(){
6
7          parent::__construct();
8
9          $this->load->model("villages_model");
10     }
11
12     public function index(){
13
14         redirect(base_url());
15     }
16
17     public function player_default_active_village(){
18
19
20         $username = $this->input->post("username");
21
22         $idvillage = $this->villages_model->
23         default_active_village($username);
24
25         $this->output->
26         set_content_type("application/json")->
27         set_output(json_encode($idvillage));
28     }
29
30
31     public function player_village_ranking(){
32
33         $idvillage = $this->input->
34         post("idvillage");
35
36         $village_ranking = $this->villages_model->
37         village_ranking($idvillage);
38
39         $this->output->
40         set_content_type("application/json")->
41         set_output(json_encode($village_ranking));
42     }
43
44     public function player_villages_rankings(){
45
46         $villages_rankings = $this->villages_model->
47         villages_rankings();
48
49         $this->output->
50         set_content_type("application/json")->
51         set_output(json_encode($villages_rankings));
```

```

52     }
53
54     public function player_villages_locations(){
55         $idvillage = $this->input->
56         post("idvillage");
57
58         $villages_locations = $this->villages_model-
59         >villages_locations($idvillage);
60
61         $this->output->
62         set_content_type("application/json")->
63         set_output(json_encode
64         ($villages_locations));
65     }
66
67     public function player_village_location(){
68
69         $idvillage = $this->input->
70         post("idvillage");
71
72         $targetvillage = $this->input->
73         post("targetvillage");
74
75         $villages_locations = $this->villages_model-
76         >village_location
77         ($idvillage,$targetvillage);
78
79         $this->output->
80         set_content_type("application/json")->
81         set_output(json_encode
82         ($villages_locations));
83     }
84
85 }
86
87 SELESAI

```

Penjelasan dari kelas Villages pada tabel 4.112 yaitu:

1. Merupakan turunan dari kelas CI_controller.
2. Berisi berbagai *method* terkait objek desa pemain yang nantinya bisa diakses oleh *game client*.

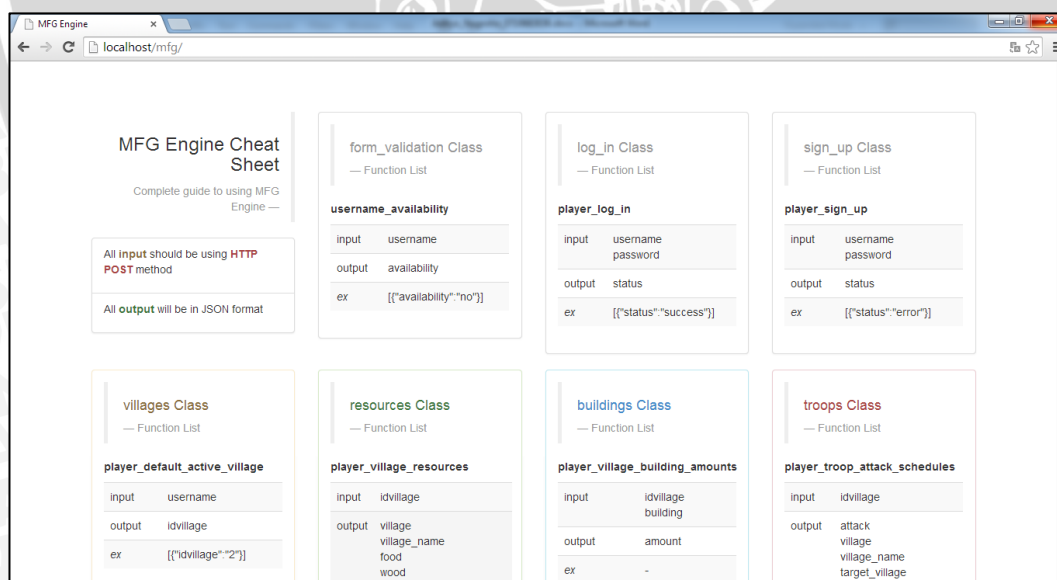
3. Kinerja semua *method* secara garis besar ialah menerima masukan dari *game client* untuk diteruskan ke *function* yang ada pada kelas *Villages_model*.
4. Masukan diterima *function* dari metode POST dengan menggunakan *function* post kelas Input Code Igniter.
5. Bilamana ada nilai keluaran untuk *game client*, *function* akan memberi keluaran dalam format JSON menggunakan kelas Output Code Igniter.
6. Untuk mengubah nilai balik dari *function-function* kelas *Villages_model* menjadi format JSON digunakan *function* *json_encode*.

4.4.3 Implementasi View

Hanya terdapat 1 *view* pada aplikasi *web game server online* permainan *strategy* Kerajaan Majapahit ialah sebagai berikut.

4.4.3.1 Implementasi Halaman Home

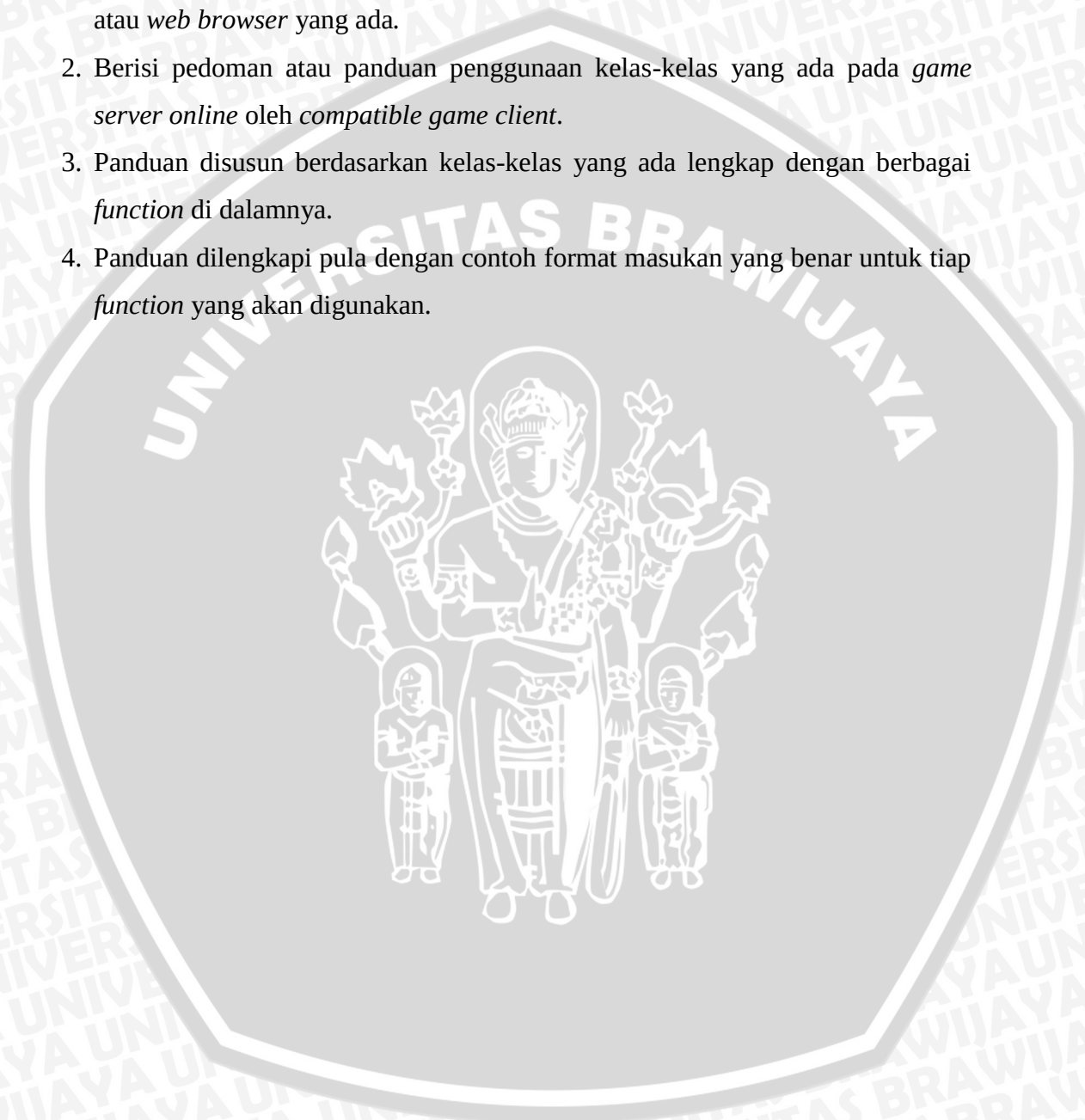
Merupakan halaman *website* yang bisa ditampilkan manakala terdapat pemesanan oleh *game client* melalui kelas *controller* Home.



Gambar 4.1 Halaman Web Home

Penjelasan dari halaman Home pada gambar 4.1 yakni:

1. Halaman *website* telah menggunakan teknologi *Responsive Web Design* yang bertujuan untuk menyediakan tampilan yang optimal untuk berbagai *device* atau *web browser* yang ada.
2. Berisi pedoman atau panduan penggunaan kelas-kelas yang ada pada *game server online* oleh *compatible game client*.
3. Panduan disusun berdasarkan kelas-kelas yang ada lengkap dengan berbagai *function* di dalamnya.
4. Panduan dilengkapi pula dengan contoh format masukan yang benar untuk tiap *function* yang akan digunakan.



BAB V

PENGUJIAN DAN ANALISIS

Dalam bab ini dibahas mengenai berbagai tahapan pengujian dan analisis dalam pengembangan *game server online* permainan *strategy* Kerajaan Majapahit yang telah diimplementasikan. Proses pengujian dalam bab ini akan dilakukan melalui tiga tahapan yakni pengujian unit, pengujian integrasi serta pengujian validasi. Pada tahapan pengujian unit dan integrasi akan digunakan teknik pengujian *White Box* (*White Box Testing*). Sedangkan pada pengujian validasi akan digunakan teknik pengujian *Black Box* (*Black Box Testing*).

5.1 Pengujian Unit

Pada pengujian unit akan digunakan metode *White Box Testing* dengan pemakaian teknik *Basis Path Testing*. Dalam teknik *Basis Path Testing*, proses pengujian dilakukan dengan cara memodelkan algoritma perangkat lunak pada sebuah *flow graph* dan kemudian dilakukan penghitungan atau kalkulasi untuk menentukan nilai *cyclometric complexity* algoritma serta melakukan uji kasus terhadap setiap *basis path set* yang dimungkinkan.

Beberapa pengujian unit yang akan dicantumkan dalam subbab ini hanya untuk algoritma dari beberapa *function* saja, sehingga tidak seluruh pengujian unit akan dicantumkan. Adapun beberapa pengujian unit yang dicantumkan ialah:

5.1.1 Pengujian Unit Pada *Function* Dalam Perangkat Lunak

1. Pengujian Unit pada *Function* `get_attribute_id`

Tabel 5.1 Pemodelan *Flow Graph* Algoritma *Function* `get_attribute_id`

Pseudocode <i>Function</i> <code>get_attribute_id</code>
Deklarasi awal <code>var_attribute_name:VARCHAR(45) -> Masukan dari user/sistem</code> <code>var_ida:INT</code>

Deskripsi

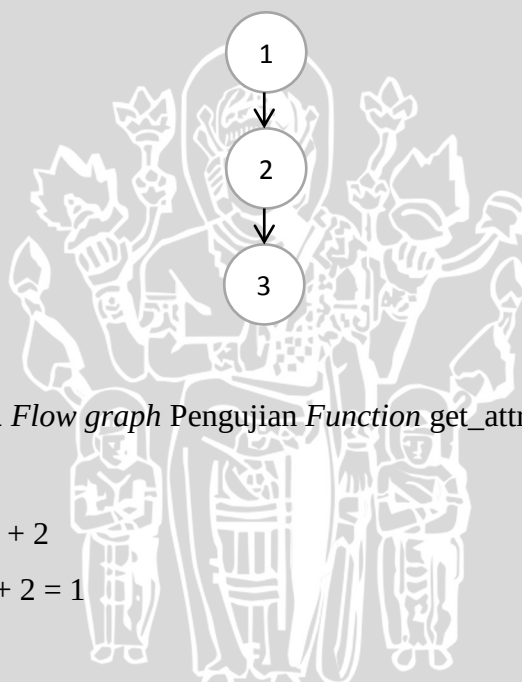
MULAI ← 1

```
SELECT ida INTO var_ida
FROM attributes
WHERE attribute = var_attribute_name;
```

2

RETURN var_ida;

SELESAI ← 3



Gambar 5.1 Flow graph Pengujian Function *get_attribute_id*

$$\begin{aligned}
 V(G) &= E - N + 2 \\
 &= 2 - 3 + 2 = 1
 \end{aligned}$$

Berdasarkan nilai *cyclomatic complexity* yang telah diperoleh dari hasil kalkulasi, maka bisa ditentukan satu buah *basis set* dari jalur *independent*, yakni :

Jalur 1 : 1 – 2 – 3

Penjelasan kasus uji pengujian *function get_attribute_id* akan dipaparkan pada tabel 5.2

Tabel 5.2 Kasus Uji pada Pengujian *Function* *get_attribute_id*

Jalur	Kasus Uji	Hasil yang diharapkan	Hasil yang didapatkan
1	Menjalankan <i>function</i> dengan masukan nama atribut yang telah terdaftar sebelumnya.	<i>Function</i> <i>get_attribute_id</i> memberikan keluaran berupa ID atribut yang memiliki nama atribut sama dengan masukan.	<i>Function</i> <i>get_attribute_id</i> memberikan keluaran berupa ID atribut yang memiliki nama atribut sama dengan masukan.

2. Pengujian Unit pada *Function* *get_datetime_variables_from_id*

Tabel 5.3 Pemodelan *Flow Graph* Algoritma *Function* *get_datetime_variables_from_id*

Pseudocode <i>Function</i> <i>get_datetime_variables_from_id</i>	
Deklarasi awal	
var_iddv:INT -> Masukan dari user/sistem	
var_value:DATETIME	
Deskripsi	
MULAI ← 1	
SELECT value INTO var_value FROM mem_datetime_variables WHERE iddv = var_iddv;	} 2
RETURN var_value;	
SELESAI ← 3	



Gambar 5.2 Flow graph Pengujian Function *get_datetime_variables_from_id*

$$\begin{aligned}
 V(G) &= E - N + 2 \\
 &= 2 - 3 + 2 = 1
 \end{aligned}$$

Berdasarkan nilai *cyclomatic complexity* yang telah diperoleh dari hasil kalkulasi, maka bisa ditentukan satu buah *basis set* dari jalur *independent*, yakni :

Jalur 1 : 1 – 2 – 3

Penjelasan kasus uji pengujian *function get_datetime_variables_from_id* akan dipaparkan pada tabel 5.4

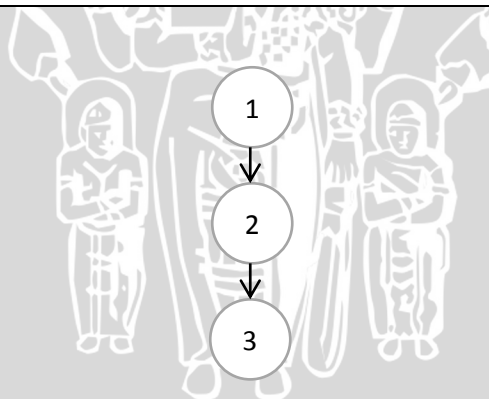
Tabel 5.4 Kasus Uji pada Pengujian Function *get_datetime_variables_from_id*

Jalur	Kasus Uji	Hasil yang diharapkan	Hasil yang didapatkan
1	Menjalankan <i>function</i> dengan masukan nilai ID yang telah terdaftar sebelumnya.	<i>Function get_datetime_variables_from_id</i> memberikan keluaran berupa nilai variabel <i>datetime</i> yang memiliki ID sama dengan masukan.	<i>Function get_datetime_variables_from_id</i> memberikan keluaran berupa nilai variabel <i>datetime</i> yang memiliki ID sama dengan masukan.

3. Pengujian Unit pada Function get_troop_id

Tabel 5.5 Pemodelan *Flow Graph* Algoritma *Function get_troop_id*

Pseudocode Function get_troop_id	
Deklarasi awal	
var_troop:VARCHAR(45) -> Masukan dari user/sistem	
var_idt:INT	
Deskripsi	
MULAI ← 1	
SELECT idt INTO var_idt FROM troops WHERE troop = var_idt;	} 2
RETURN var_idt;	
SELESAI ← 3	



Gambar 5.3 *Flow graph* Pengujian *Function get_troop_id*

$$\begin{aligned}
 V(G) &= E - N + 2 \\
 &= 2 - 3 + 2 = 1
 \end{aligned}$$

Berdasarkan nilai *cyclomatic complexity* yang telah diperoleh dari hasil kalkulasi, maka bisa ditentukan satu buah *basis set* dari jalur *independent*, yakni :

Jalur 1 : 1 – 2 – 3

Penjelasan kasus uji pengujian *function* *get_troop_id* akan dipaparkan pada tabel 5.6

Tabel 5.6 Kasus Uji pada Pengujian *Function* *get_troop_id*

Jalur	Kasus Uji	Hasil yang diharapkan	Hasil yang didapatkan
1	Menjalankan <i>function</i> dengan masukan nama pasukan yang telah terdaftar sebelumnya.	<i>Function</i> <i>get_troop_id</i> memberikan keluaran berupa ID pasukan yang memiliki nama pasukan sama dengan masukan.	<i>Function</i> <i>get_troop_id</i> memberikan keluaran berupa ID pasukan yang memiliki nama pasukan sama dengan masukan.

4. Pengujian Unit pada *Function* *get_new_attack*

Tabel 5.7 Pemodelan *Flow Graph* Algoritma *Function* *get_new_attack*

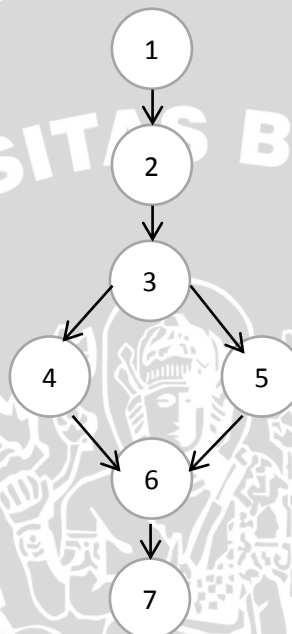
Pseudocode <i>Function</i> <i>get_new_attack</i>	
Deklarasi awal	
var_attack:INT	
Deskripsi	
MULAI ← 1	
SELECT MAX(attack) INTO var_attack FROM villages_troops_attacks LOCK IN SHARE MODE;	} 2
IF var_attack IS NULL ← 3 THEN	
SET var_attack = 0; ← 4	
ELSE	
SET var_attack = var_attack; ← 5	
END IF;	

```

SET var_attack = var_attack + 1;
RETURN var_attack;
SELESAI ← 7

```

} 6



Gambar 5.4 Flow graph Pengujian Function *get_new_attack*

$$\begin{aligned}
 V(G) &= E - N + 2 \\
 &= 7 - 7 + 2 = 2
 \end{aligned}$$

Berdasarkan nilai *cyclomatic complexity* yang telah diperoleh dari hasil kalkulasi, maka bisa ditentukan dua buah basis set dari jalur *independent*, yakni :

Jalur 1 : 1 – 2 – 3 – 4 – 6 – 7

Jalur 2 : 1 – 2 – 3 – 5 – 6 – 7

Penjelasan kasus uji pengujian *function* *get_new_attack* akan dipaparkan pada tabel 5.8

Tabel 5.8 Kasus Uji pada Pengujian *Function* get_new_attack

Jalur	Kasus Uji	Hasil yang diharapkan	Hasil yang didapatkan
1	Menjalankan <i>function</i> ketika belum terdapat data sama sekali dalam tabel villages_troops_attacks.	<i>Function</i> get_new_attack memberikan nilai keluaran 1 pada penggunaan <i>function</i> get_new_attack.	<i>Function</i> get_new_attack memberikan nilai keluaran 1 pada penggunaan <i>function</i> get_new_attack.
2	Menjalankan <i>function</i> ketika sudah terdapat data dalam tabel villages_troops_attacks.	<i>Function</i> get_new_attack memberikan nilai keluaran berupa hasil dari nilai maksimal yang ditemukan pada kolom attack ditambah dengan nilai 1 pada penggunaan <i>function</i> get_new_attack.	<i>Function</i> get_new_attack memberikan nilai keluaran berupa hasil dari nilai maksimal yang ditemukan pada kolom attack ditambah dengan nilai 1 pada penggunaan <i>function</i> get_new_attack.

5. Pengujian Unit pada Function get_new_idv

Tabel 5.9 Pemodelan *Flow Graph* Algoritma *Function* get_new_idv

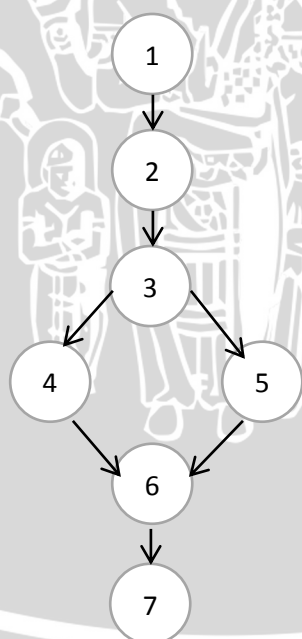
Pseudocode Function get_new_idv
Deklarasi awal var_idv:INT Deskripsi MULAI ← 1


```
SELECT MAX(idv)
INTO var_idv
FROM villages LOCK IN SHARE MODE; } 2

IF var_idv IS NULL ← 3
THEN
    SET var_idv = 0; ← 4
ELSE
    SET var_idv = var_idv; ← 5
END IF;

SET var_idv = var_idv + 1;
RETURN var_idv; } 6

SELESAI ← 7
```



Gambar 5.5 Flow graph Pengujian Function *get_new_idv*

$$\begin{aligned} V(G) &= E - N + 2 \\ &= 7 - 7 + 2 = 2 \end{aligned}$$

Berdasarkan nilai *cyclomatic complexity* yang telah diperoleh dari hasil kalkulasi, maka bisa ditentukan dua buah basis set dari jalur *independent*, yakni :

Jalur 1 : 1 – 2 – 3 – 4 – 6 – 7

Jalur 2 : 1 – 2 – 3 – 5 – 6 – 7

Penjelasan kasus uji pengujian *function get_new_idv* akan dipaparkan pada tabel 5.10

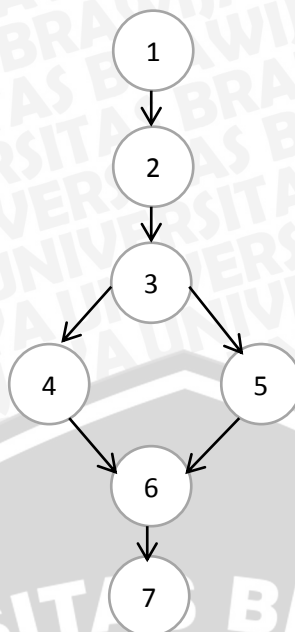
Tabel 5.10 Kasus Uji pada Pengujian *Function get_new_idv*.

Jalur	Kasus Uji	Hasil yang diharapkan	Hasil yang didapatkan
1	Menjalankan <i>function</i> ketika belum terdapat data sama sekali dalam tabel <i>villages</i> .	<i>Function get_new_idv</i> memberikan nilai keluaran 1 pada penggunaan <i>function get_new_idv</i> .	<i>Function get_new_idv</i> memberikan nilai keluaran 1 pada penggunaan <i>function get_new_idv</i> .
2	Menjalankan <i>function</i> ketika sudah terdapat data dalam tabel <i>villages</i> .	<i>Function get_new_idv</i> memberikan nilai keluaran berupa hasil dari nilai maksimal yang ditemukan pada kolom <i>idv</i> ditambah dengan nilai 1 pada penggunaan <i>function get_new_idv</i> .	<i>Function get_new_idv</i> memberikan nilai keluaran berupa hasil dari nilai maksimal yang ditemukan pada kolom <i>idv</i> ditambah dengan nilai 1 pada penggunaan <i>function get_new_idv</i> .

6. Pengujian Unit pada Function get_new_idvas

Tabel 5.11 Pemodelan *Flow Graph* Algoritma Function get_new_idvas

Pseudocode Function get_new_idvas	
Deklarasi awal	
var_idvas:INT	
Deskripsi	
MULAI ← 1	
SELECT MAX(idvas) INTO var_idvas FROM villages_resources_schedules LOCK IN SHARE MODE;	} 2
IF var_idvas IS NULL ← 3 THEN	
SET var_idvas = 0; ← 4	
ELSE	
SET var_idvas = var_idvas; ← 5	
END IF;	
SET var_idvas = var_idvas + 1;	} 6
RETURN var_idvas;	
SELESAI ← 7	



Gambar 5.6 Flow graph Pengujian Function *get_new_idvas*

$$\begin{aligned}
 V(G) &= E - N + 2 \\
 &= 7 - 7 + 2 = 2
 \end{aligned}$$

Berdasarkan nilai *cyclomatic complexity* yang telah diperoleh dari hasil kalkulasi, maka bisa ditentukan dua buah basis set dari jalur *independent*, yakni :

Jalur 1 : 1 – 2 – 3 – 4 – 6 – 7

Jalur 2 : 1 – 2 – 3 – 5 – 6 – 7

Penjelasan kasus uji pengujian *function get_new_idvas* akan dipaparkan pada tabel 5.12

Tabel 5.12 Kasus Uji untuk Pengujian Function *get_new_idvas*

Jalur	Kasus Uji	Hasil yang diharapkan	Hasil yang didapatkan
1	Menjalankan <i>function</i> ketika belum terdapat data sama sekali dalam tabel	<i>Function get_new_idvas</i> memberikan nilai keluaran 1 pada penggunaan <i>function get_new_idvas</i> .	<i>Function get_new_idvas</i> memberikan nilai keluaran 1 pada penggunaan <i>function get_new_idvas</i> .

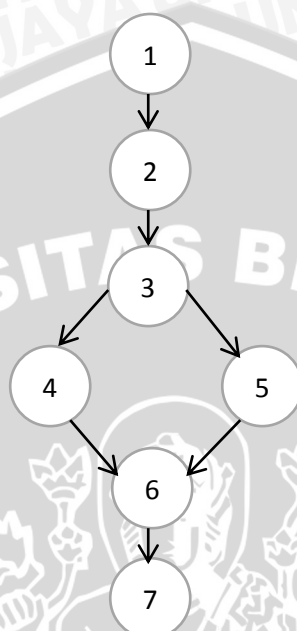
	villages_resource_schedules.		
2	Menjalankan <i>function</i> ketika sudah terdapat data dalam tabel villages_resource_schedules.	<i>Function</i> get_new_idvas memberikan nilai keluaran berupa hasil dari nilai maksimal yang ditemukan pada kolom idvas ditambah dengan nilai 1 pada penggunaan <i>function</i> get_new_idvas.	<i>Function</i> get_new_idvas memberikan nilai keluaran berupa hasil dari nilai maksimal yang ditemukan pada kolom idvas ditambah dengan nilai 1 pada penggunaan <i>function</i> get_new_idvas.

7. Pengujian Unit pada Function get_new_idvt

Tabel 5.13 Pemodelan Flow Graph Algoritma Function get_new_idvt

Pseudocode Function get_new_idvt	
Deklarasi awal	
var_idvt:INT	
Deskripsi	
MULAI ← 1	
SELECT MAX(idvt) INTO var_idvt FROM villages_troops LOCK IN SHARE MODE;	} 2
IF var_idvt IS NULL ← 3 THEN	
SET var_idvt = 0; ← 4	
ELSE	
SET var_idvt = var_idvt; ← 5	
END IF;	
SET var_idvt = var_idvt + 1;	} 6
RETURN var_idvt;	

SELESAI ← 7



Gambar 5.7 Flow graph Pengujian Function *get_new_idvt*

$$\begin{aligned}
 V(G) &= E - N + 2 \\
 &= 7 - 7 + 2 = 2
 \end{aligned}$$

Berdasarkan nilai *cyclomatic complexity* yang telah diperoleh dari hasil kalkulasi, maka bisa ditentukan dua buah basis set dari jalur *independent*, yakni :

Jalur 1 : 1 – 2 – 3 – 4 – 6 – 7

Jalur 2 : 1 – 2 – 3 – 5 – 6 – 7

Penjelasan kasus uji pengujian *function get_new_idvt* akan dipaparkan pada tabel 5.14

Tabel 5.14 Kasus Uji untuk Pengujian *Function* *get_new_idvt*

Jalur	Kasus Uji	Hasil yang diharapkan	Hasil yang didapatkan
1	Menjalankan <i>function</i> ketika belum terdapat data sama sekali dalam tabel <i>villages_troops</i> .	<i>Function</i> <i>get_new_idvt</i> memberikan nilai keluaran 1 pada penggunaan <i>function</i> <i>get_new_idvt</i> .	<i>Function</i> <i>get_new_idvt</i> memberikan nilai keluaran 1 pada penggunaan <i>function</i> <i>get_new_idvt</i> .
2	Menjalankan <i>function</i> ketika sudah terdapat data dalam tabel <i>villages_troops</i> .	<i>Function</i> <i>get_new_idvt</i> memberikan nilai keluaran berupa hasil dari nilai maksimal yang ditemukan pada kolom <i>idvt</i> ditambah dengan nilai 1 pada penggunaan <i>function</i> <i>get_new_idvt</i> .	<i>Function</i> <i>get_new_idvt</i> memberikan nilai keluaran berupa hasil dari nilai maksimal yang ditemukan pada kolom <i>idvt</i> ditambah dengan nilai 1 pada penggunaan <i>function</i> <i>get_new_idvt</i> .

8. Pengujian Unit pada *Function* *get_varchar_variables_id*

Tabel 5.15 Pemodelan *Flow Graph* Algoritma *Function* *get_varchar_variables_id*

Pseudocode <i>Function</i> <i>get_varchar_variables_id</i>	
Deklarasi awal	
<pre>var_variable:VARCHAR(45) -> Masukan dari user/sistem var_idvv:INT</pre>	
Deskripsi	
MULAI ← 1	
<pre>SELECT idvv INTO var_idvv FROM mem_varchar_variables WHERE variable = var_variable;</pre>	2
RETURN <i>var_idvv</i> ;	

SELESAI ←

3



Gambar 5.8 Flow graph Pengujian Function `get_varchar_variables_id`

Pemodelan ke dalam bentuk *flow graph* (Gambar 5.1) yang telah dilakukan terhadap *function* `get_varchar_variables_id` menghasilkan jumlah kompleksitas siklomatis (*cyclomatic complexity*) yang dapat dikalkulasi melalui rumus atau persamaan $V(G) = E - N + 2$, dimana $V(G)$ merupakan jumlah kompleksitas siklomatis, E merupakan sisi atau *edge* (garis penghubung antar *node*) dan N merupakan jumlah dari simpul (*node*).

$$\begin{aligned} V(G) &= E - N + 2 \\ &= 2 - 3 + 2 = 1 \end{aligned}$$

Berdasarkan nilai *cyclomatic complexity* yang telah diperoleh dari hasil kalkulasi, maka bisa ditentukan satu buah *basis set* dari jalur *independent*, yakni :

Jalur 1 : 1 – 2 – 3

Penjelasan kasus uji pengujian *function* `get_varchar_variables_id` akan dipaparkan pada (Tabel 5.16)

Tabel 5.16 Kasus Uji pada Pengujian *Function* *get_varchar_variables_id*

Jalur	Kasus Uji	Hasil yang diharapkan	Hasil yang didapatkan
1	Menjalankan <i>function</i> dengan masukan nama variabel yang telah terdaftar sebelumnya.	<i>Function</i> <i>get_varchar_variables_id</i> memberikan keluaran berupa ID variabel yang memiliki nama variabel sama dengan masukan.	<i>Function</i> <i>get_varchar_variables_id</i> memberikan keluaran berupa ID variabel yang memiliki nama variabel sama dengan masukan.

5.1.2 Analisis pengujian Unit

Hasil analisis yang bisa diambil dari serangkaian tahapan pengujian unit yang telah dilakukan yakni:

1. Berdasarkan adanya kesesuaian antara seluruh hasil pengujian tiap unit *function* dengan *output* atau hasil keluaran yang diharapkan pada *game server online* permainan *strategy* Kerajaan Majapahit maka bisa ditarik kesimpulan bahwa unit *function* dari perangkat lunak yang telah dikembangkan sudah mampu beroperasi sebagaimana mestinya serta mampu menghasilkan *output* sesuai dengan yang diharapkan.
2. Berdasarkan hasil kalkulasi atau perhitungan *cyclomatic complexity* dari tiap *flow graph* algoritma *function* yang telah diuji, Jumlah jalur kasus uji bisa bermacam-macam bergantung dari kompleksitas algoritma *function* yang mengacu pada banyaknya seleksi kondisi. Hasil *cyclomatic complexity* terbanyak yang ditemukan saat proses pengujian unit terdapat pada *function* *get_new_attack*, *get_new_idv*, *get_new_idvas* dan *get_new_idvt*.

5.2 Pengujian Integrasi

Pengujian integrasi akan diterapkan pada *function* atau *procedure* yang mengintegrasikan fungsionalitas dari berbagai *function* atau *procedure* lain untuk melakukan suatu operasi tertentu. Pada pengujian integrasi akan digunakan

metode *White Box Testing* dengan pemakaian teknik *Basis Path Testing*. Dalam teknik *Basis Path Testing*, proses pengujian dilakukan dengan cara memodelkan algoritma perangkat lunak pada sebuah *flow graph*, melakukan penghitungan atau kalkulasi untuk menentukan nilai *cyclometric complexity* algoritma serta melakukan uji kasus terhadap setiap basis path yang dimungkinkan.

Beberapa pengujian integrasi yang akan dicantumkan dalam subbab ini hanya untuk algoritma dari beberapa *function* dan *procedure* saja, sehingga tidak seluruh pengujian integrasi akan dicantumkan. Adapun beberapa pengujian integrasi yang dicantumkan ialah:

5.2.1 Pengujian Integrasi Pada *Function* dan *Procedure* Dalam Perangkat Lunak

1. Pengujian Integrasi pada *Function* `get_village_axis`

Tabel 5.17 Pemodelan *Flow Graph* Algoritma *Function* `get_village_axis`

Pseudocode Function get_village_axis	
Deklarasi awal function get_attribute_id	
var_attribute_name:VARCHAR(45) -> Masukan dari user/sistem	
var_ida:INT	
Deskripsi function get_attribute_id	
MULAI ← 1	
SELECT ida INTO var_ida	} 2
FROM attributes	
WHERE attribute = var_attribute_name;	
RETURN var_ida;	
SELESAI ← 3	
Deklarasi awal	
var_village:INT -> Masukan dari user/sistem	

```
var_axis:VARCHAR(50) -> Masukan dari user/sistem
var_attribute:INT
var_value:INT
```

Deskripsi

MULAI ← 4

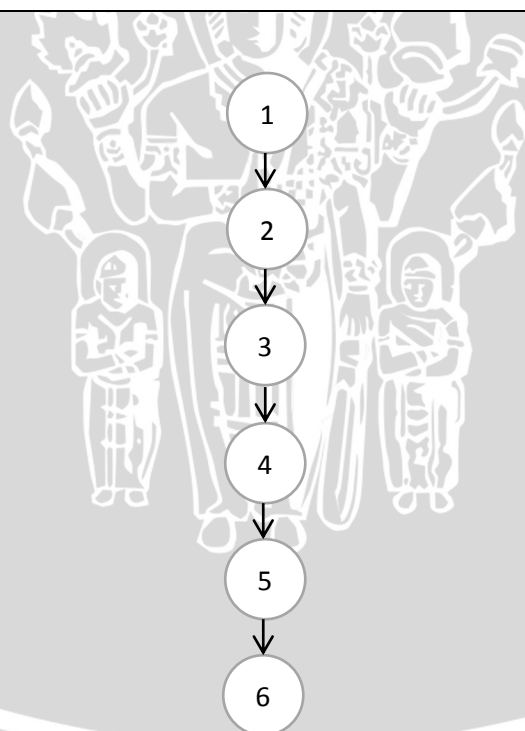
```
SET var_attribute = get_attribute_id(var_axis);
```

```
SELECT value INTO var_value
FROM villages_locations
WHERE village = var_village
AND attribute = var_attribute;
```

```
RETURN var_value;
```

SELESAI ← 6

5



Gambar 5.9 Flow graph Pengujian Function get_village_axis

$$\begin{aligned}
 V(G) &= E - N + 2 \\
 &= 5 - 6 + 2 = 1
 \end{aligned}$$

Berdasarkan nilai *cyclomatic complexity* yang telah diperoleh dari hasil kalkulasi, maka bisa ditentukan satu buah *basis set* dari jalur *independent*, yakni :

Jalur 1 : 1 – 2 – 3 – 4 – 5 – 6

Penjelasan kasus uji pengujian *function get_village_axis* akan dipaparkan pada tabel 5.18

Tabel 5.18 Kasus Uji pada Pengujian *Function get_village_axis*

Jalur	Kasus Uji	Hasil yang diharapkan	Hasil yang didapatkan
1	Menjalankan <i>function</i> dengan masukan ID desa yang telah terdaftar sebelumnya dan jenis koordinat lokasi yang ingin diketahui.	<i>Function get_village_axis</i> memberikan keluaran berupa nilai koordinat lokasi sumbu tertentu yang ingin diketahui dari masukan ID desa.	<i>Function get_village_axis</i> memberikan keluaran berupa nilai koordinat lokasi sumbu tertentu yang ingin diketahui dari masukan ID desa.

2. Pengujian Integrasi pada Procedure *get_village_location*

Tabel 5.19 Pemodelan *Flow Graph* Algoritma *Procedure get_village_location*

Pseudocode Procedure <i>get_village_location</i>
Deklarasi awal <i>function get_attribute_id</i> <i>var_attribute_name</i>:VARCHAR(45) -> Masukan dari user/sistem <i>var_ida</i>:INT Deskripsi <i>function get_attribute_id</i> MULAI ← 1


```
SELECT ida INTO var_ida
FROM attributes
WHERE attribute = var_attribute_name;
```

```
RETURN var_ida;
```

```
SELESAI ← 3
```

2

Deklarasi awal function get_village_axis

```
var_village:INT -> Masukan dari user/sistem
var_axis:VARCHAR(50) -> Masukan dari user/sistem
var_attribute:INT
var_value:INT
```

Deskripsi function get_village_axis

```
MULAI ← 4
```

```
SET var_attribute = get_attribute_id(var_axis);
```

```
SELECT value INTO var_value
FROM villages_locations
WHERE village = var_village
AND attribute = var_attribute;
```

```
RETURN var_value;
```

```
SELESAI ← 6
```

5

Deklarasi awal

```
var_village:INT -> Masukan dari user/sistem
var_x_axis:INT -> Keluaran dari sistem
var_y_axis:INT -> Keluaran dari sistem
```

Deskripsi

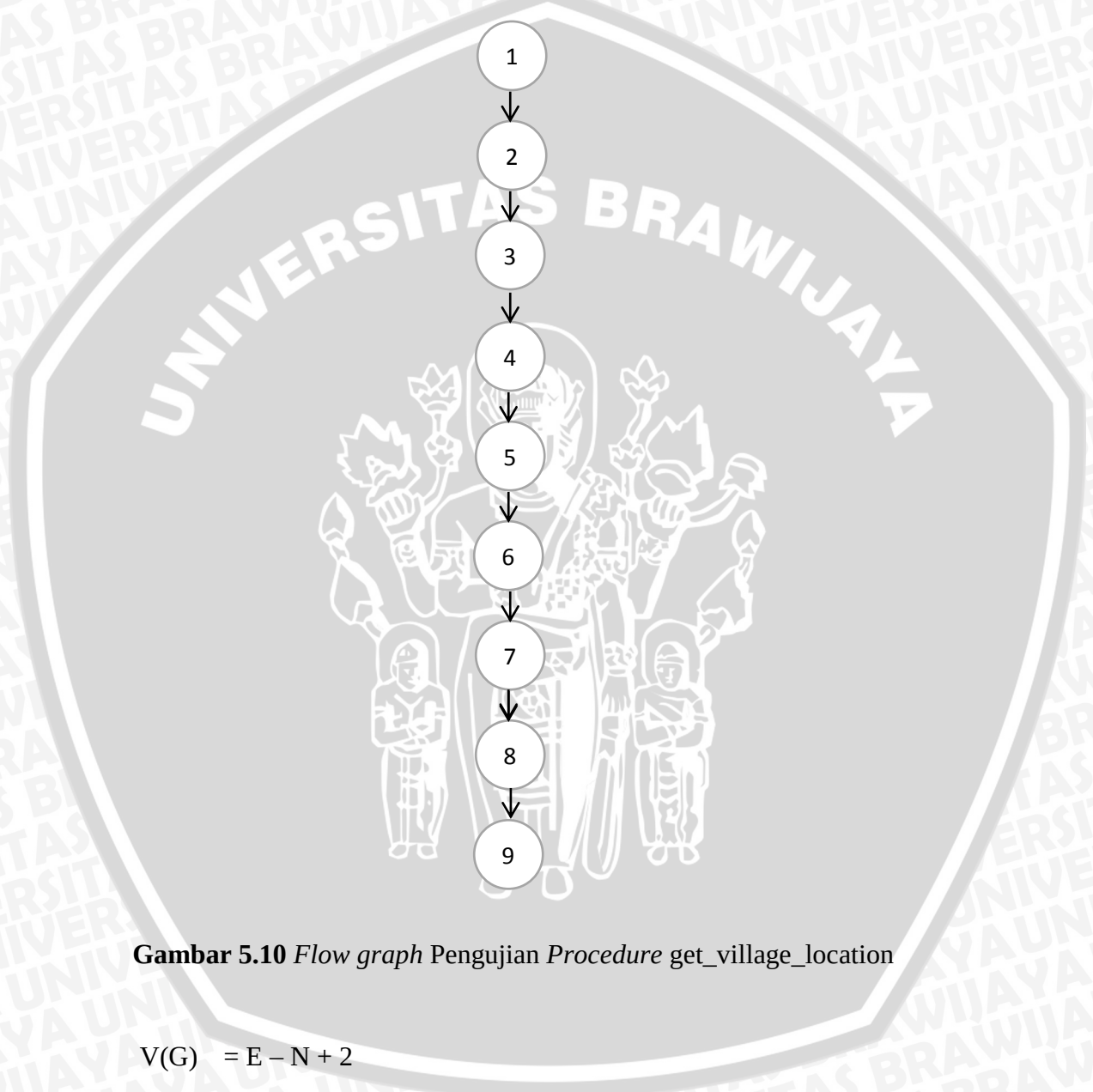
```
MULAI ← 7
```

```
SET var_x_axis =
get_village_axis(var_village,"x_axis");
```

8

```
SET var_y_axis =  
get_village_axis(var_village,"y_axis");
```

SELESAI ← 9



Gambar 5.10 Flow graph Pengujian Procedure *get_village_location*

$$\begin{aligned} V(G) &= E - N + 2 \\ &= 8 - 9 + 2 = 1 \end{aligned}$$

Berdasarkan nilai *cyclomatic complexity* yang telah diperoleh dari hasil kalkulasi, maka bisa ditentukan satu buah *basis set* dari jalur *independent*, yakni :

Jalur 1 : 1-2-3-4-5-6-7-8-9

Penjelasan kasus uji pengujian *procedure* *get_village_location* akan dipaparkan pada tabel 5.20

Tabel 5.20 Kasus Uji pada Pengujian *Procedure* *get_village_location*

Jalur	Kasus Uji	Hasil yang diharapkan	Hasil yang didapatkan
1	Menjalankan <i>procedure</i> dengan masukan ID desa yang telah terdaftar sebelumnya.	<i>Procedure</i> <i>get_village_location</i> memberikan keluaran berupa nilai koordinat lokasi sumbu x dan sumbu y dari masukan ID desa.	<i>Procedure</i> <i>get_village_location</i> memberikan keluaran berupa nilai koordinat lokasi sumbu x dan sumbu y dari masukan ID desa.

3. Pengujian Integrasi pada Function *get_distance*

Tabel 5.21 Pemodelan *Flow Graph* Algoritma *Function* *get_distance*

Pseudocode Procedure <i>get_distance</i>	
Deklarasi awal function <i>get_attribute_id</i>	
var_attribute_name:VARCHAR(45) -> Masukan dari user/sistem	
var_ida:INT	
Deskripsi function <i>get_attribute_id</i>	
MULAI ← 1	
SELECT ida INTO var_ida FROM attributes WHERE attribute = var_attribute_name;	2
RETURN var_ida;	
SELESAI ← 3	

Deklarasi awal function get_village_axis

```
var_village:INT -> Masukan dari user/sistem
var_axis:VARCHAR(50) -> Masukan dari user/sistem
var_attribute:INT
var_value:INT
```

Deskripsi function get_village_axis

MULAI ← 4

```
SET var_attribute = get_attribute_id(var_axis);
```

```
SELECT value INTO var_value
FROM villages_locations
WHERE village = var_village
AND attribute = var_attribute;
```

5

```
RETURN var_value;
```

SELESAI ← 6

Deklarasi awal procedure get_village_location

```
var_village:INT -> Masukan dari user/sistem
var_x_axis:INT -> Keluaran dari sistem
var_y_axis:INT -> Keluaran dari sistem
```

Deskripsi procedure get_village_location

MULAI ← 7

```
SET var_x_axis =
get_village_axis(var_village,"x_axis");
```

```
SET var_y_axis =
get_village_axis(var_village,"y_axis");
```

8

SELESAI ← 9

Deklarasi awal

```
var_village:INT -> Masukan dari user/sistem
var_target_village:INT -> Masukan dari user/sistem
```

```
var_village_x:INT  
var_village_y:INT  
var_target_village_x:INT  
var_target_village_y:INT  
var_distance:FLOAT
```

Deskripsi

MULAI ← 10

```
CALL get_village_location  
(var_village,var_village_x,var_village_y);
```

```
CALL get_village_location  
(var_target_village,var_target_village_x,  
var_target_village_y);
```

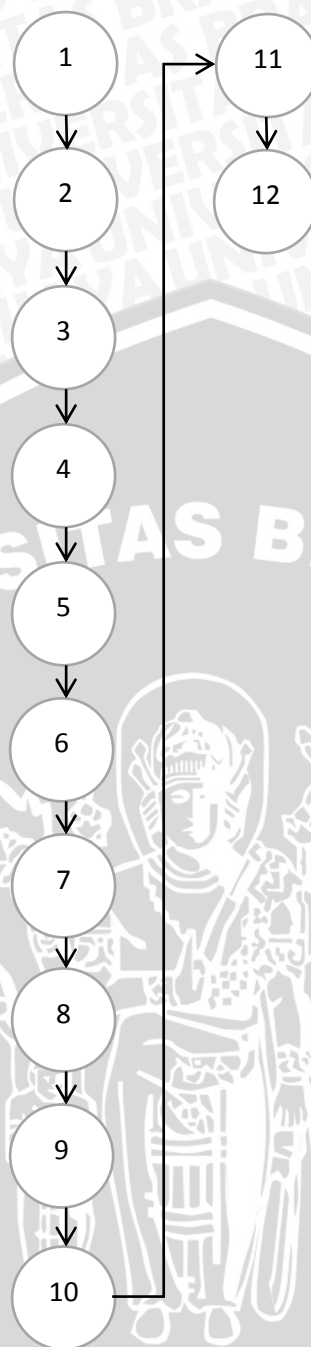
```
SET var_distance = SQRT(POW(var_target_village_x-  
var_village_x,2)+POW(var_target_village_y-  
var_village_y,2));
```

```
RETURN var_distance;
```

SELESAI ← 12

11





Gambar 5.11 Flow graph Pengujian Function *get_distance*

$$\begin{aligned}
 V(G) &= E - N + 2 \\
 &= 11 - 12 + 2 = 1
 \end{aligned}$$

Berdasarkan nilai *cyclomatic complexity* yang telah diperoleh dari hasil kalkulasi, maka bisa ditentukan satu buah *basis set* dari jalur *independent*, yakni :

Jalur 1 : 1 – 2 – 3 – 4 – 5 – 6 – 7 – 8 – 9 – 10 – 11 – 12

Penjelasan kasus uji pengujian *function* *get_distance* akan dipaparkan pada tabel 5.22

Tabel 5.22 Kasus Uji pada Pengujian *function* *get_distance*

Jalur	Kasus Uji	Hasil yang diharapkan	Hasil yang didapatkan
1	Menjalankan <i>function</i> dengan masukan ID desa dan ID target desa yang telah terdaftar sebelumnya.	<i>function</i> <i>get_distance</i> memberikan keluaran berupa nilai jarak antara desa pemain dengan desa target serangan.	<i>function</i> <i>get_distance</i> memberikan keluaran berupa nilai jarak antara desa pemain dengan desa target serangan.

4. Pengujian Integrasi pada Function *get_travel_time*

Tabel 5.23 Pemodelan *Flow Graph* Algoritma *Function* *get_travel_time*

Pseudocode Procedure <i>get_travel_time</i>	
Deklarasi awal <i>function</i> <i>get_attribute_id</i>	
var_attribute_name:VARCHAR(45) -> Masukan dari user/sistem	
var_ida:INT	
Deskripsi <i>function</i> <i>get_attribute_id</i>	
MULAI ← 1	
SELECT ida INTO var_ida FROM attributes WHERE attribute = var_attribute_name;	2
RETURN var_ida;	

SELESAI ← 3

Deklarasi awal function get_village_axis

```
var_village:INT -> Masukan dari user/sistem
var_axis:VARCHAR(50) -> Masukan dari user/sistem
var_attribute:INT
var_value:INT
```

Deskripsi function get_village_axis

MULAI ← 4

```
SET var_attribute = get_attribute_id(var_axis);
```

```
SELECT value INTO var_value
FROM villages_locations
WHERE village = var_village
AND attribute = var_attribute;
RETURN var_value;
```

5

SELESAI ← 6

Deklarasi awal procedure get_village_location

```
var_village:INT -> Masukan dari user/sistem
var_x_axis:INT -> Keluaran dari sistem
var_y_axis:INT -> Keluaran dari sistem
```

Deskripsi procedure get_village_location

MULAI ← 7

```
SET var_x_axis =
get_village_axis(var_village,"x_axis");
```

```
SET var_y_axis =
get_village_axis(var_village,"y_axis");
```

8

SELESAI ← 9

Deklarasi awal function get_distance

```
var_village:INT -> Masukan dari user/sistem
var_target_village:INT -> Masukan dari user/sistem
var_village_x:INT
var_village_y:INT
var_target_village_x:INT
var_target_village_y:INT
var_distance:FLOAT
```

Deskripsi function get_distance

MULAI ← 10

```
CALL get_village_location
(var_village,var_village_x,var_village_y);
```

```
CALL get_village_location
(var_target_village,var_target_village_x,
var_target_village_y);
```

11

```
SET var_distance = SQRT(POW(var_target_village_x-
var_village_x,2)+POW(var_target_village_y-
var_village_y,2));
```

```
RETURN var_distance;
```

SELESAI ← 12

Deklarasi awal

```
var_troop:INT -> Masukan dari user/sistem
var_village:INT -> Masukan dari user/sistem
var_target_village:INT -> Masukan dari user/sistem
var_travel_time:TIME
var_distance:INT
var_speed:INT
```

Deskripsi

MULAI ← 13

```
SET var_distance =
get_distance(var_village,var_target_village);
```

14

```
SELECT iv.value INTO var_speed
FROM troops_stats ts
LEFT JOIN mem_integer_variables iv
ON ts.value iv = iv.idiv
```



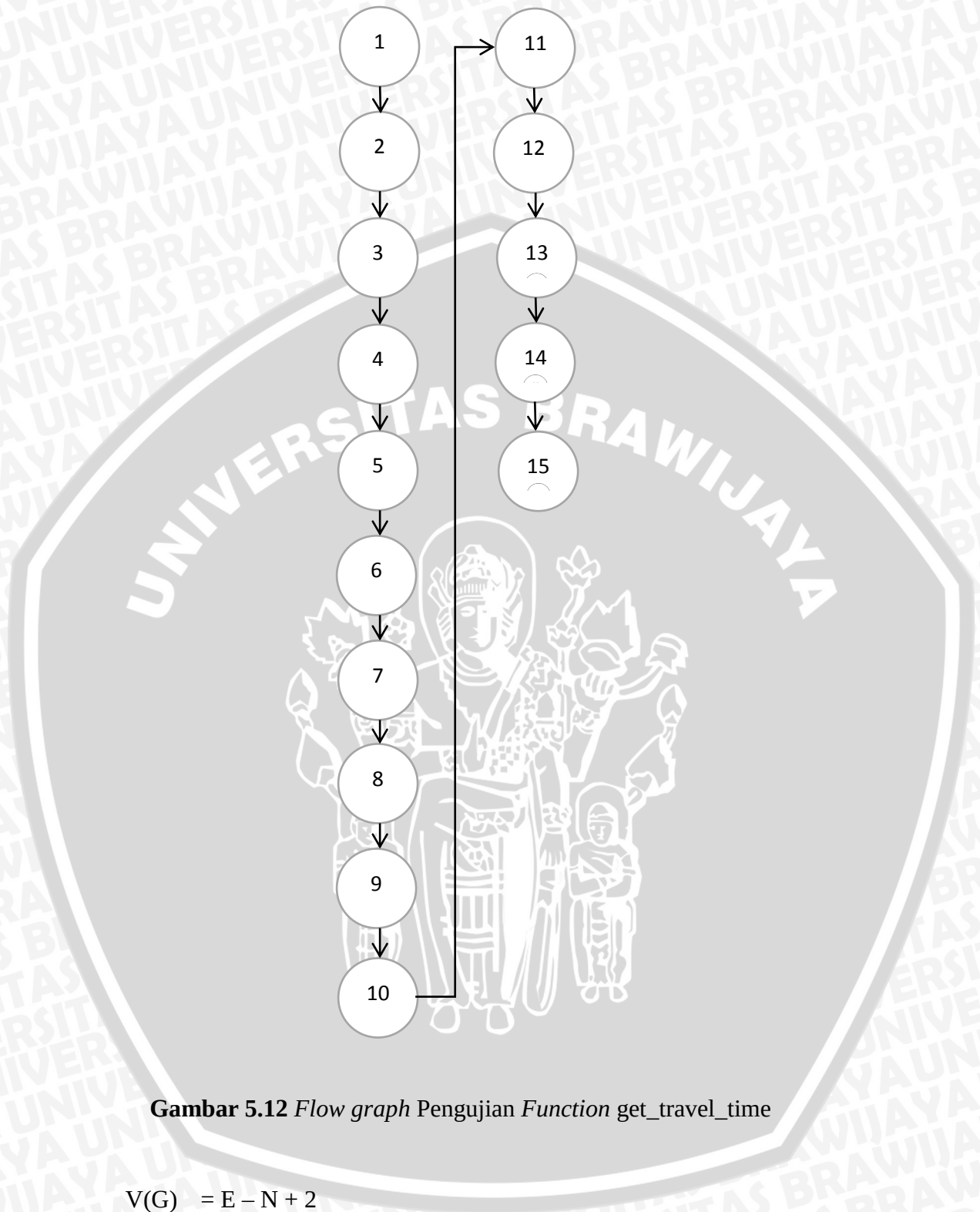
```
LEFT JOIN attributes a ON ts.attribute = a.ida  
WHERE a.attribute = "troop_speed"  
AND ts.troop = var_troop;
```

```
SET var_travel_time := SEC_TO_TIME  
(var_distance / var_speed * 3600);
```

```
RETURN var_travel_time;
```

```
SELESAI ← 15
```





Gambar 5.12 Flow graph Pengujian Function *get_travel_time*

$$\begin{aligned}
 V(G) &= E - N + 2 \\
 &= 11 - 12 + 2 = 1
 \end{aligned}$$

Berdasarkan nilai *cyclomatic complexity* yang telah diperoleh dari hasil kalkulasi, maka bisa ditentukan satu buah *basis set* dari jalur *independent*, yakni :

Jalur 1 : 1 – 2 – 3 – 4 – 5 – 6 – 7 – 8 – 9 – 10 – 11 – 12 – 13 – 14 – 15

Penjelasan kasus uji pengujian *function* *get_travel_time* akan dipaparkan pada tabel 5.24

Tabel 5.24 Kasus Uji pada Pengujian *function* *get_travel_time*

Jalur	Kasus Uji	Hasil yang diharapkan	Hasil yang didapatkan
1	Menjalankan <i>function</i> dengan masukan ID pasukan, ID desa dan ID target desa yang telah terdaftar sebelumnya.	<i>function</i> <i>get_travel_time</i> memberikan keluaran berupa nilai waktu tempuh antara desa pemain dengan desa target serangan.	<i>function</i> <i>get_travel_time</i> memberikan keluaran berupa nilai waktu tempuh antara desa pemain dengan desa target serangan.

5. Pengujian Integrasi pada Procedure *generate_village_resource_schedule_from_resources_buildings*

Tabel 5.25 Pemodelan *Flow Graph* Algoritma *Function* *generate_village_resource_schedule_from_resources_buildings*

Pseudocode Function <i>generate_village_resource_schedule_from_resources_buildings</i>
Deklarasi awal <i>function</i> <i>get_datetime_variables_from_id</i> <i>var_iddv</i>:INT -> Masukan dari user/sistem <i>var_value</i>:DATETIME Deskripsi <i>function</i> <i>get_datetime_variables_from_id</i> MULAI ← 1


```

SELECT value INTO var_value
FROM mem_datetime_variables
WHERE iddv = var_iddv;
RETURN var_value;
SELESAI ← 3

Deklarasi awal function get_new_idvas
var_idvas:INT
Deskripsi function get_new_idvas
MULAI ← 4

SELECT MAX(idvas)
INTO var_idvas
FROM villages_resources_schedules LOCK IN SHARE MODE;
IF var_idvas IS NULL
THEN
    SET var_idvas = 0;
ELSE
    SET var_idvas = var_idvas;
END IF;
SET var_idvas = var_idvas + 1;
RETURN var_idvas;
SELESAI ← 10

Deklarasi awal
var_idvb:INT -> Masukan dari user/sistem
var_idv:INT -> Masukan dari user/sistem
var_idb:INT -> Masukan dari user/sistem
var_attribute_id:INT
var_value_id:INT
var_interlude_id:INT
var_iteration_id:INT
var_execution_time:TIMESTAMP
var_idvas:INT

```

Deskripsi

MULAI ← 11

```
IF(EXISTS(SELECT idrb FROM resources_buildings
WHERE building = var_idb) = TRUE) } 12
THEN
```

```
    SELECT attribute,value_iv,interlude_dv,
    iteration_vv
    INTO
    var_attribute_id,var_value_id,var_interlude_id,
    var_iteration_id
    FROM resources_buildings WHERE building =
    var_idb;
```

```
    SET var_execution_time = NOW() +
    get_datetime_variables_from_id
    (var_interlude_id);
```

```
    SET var_idvas = get_new_idvas();
```

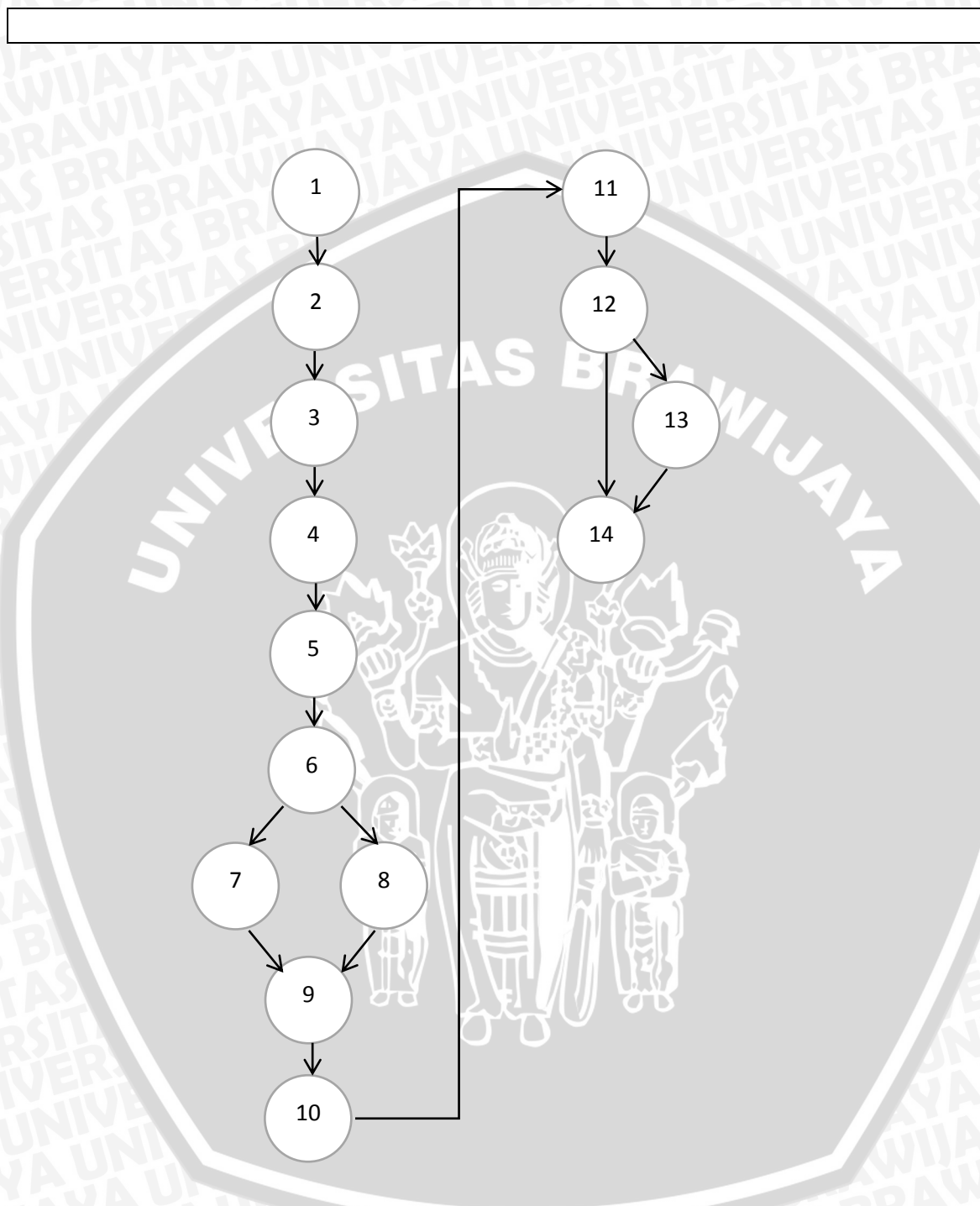
```
    INSERT
    INTO villages_resources_schedules SET
    idvas = var_idvas,
```

```
    village = var_idv,
    attribute = var_attribute_id,
    value_iv = var_value_id,
    execution_time = var_execution_time,
    interlude_dv = var_interlude_id,
    iteration_vv = var_iteration_id;
```

```
    INSERT INTO villages_buildings_resources_
    schedules SET
    village_building = var_idvb,
    village_resource_schedule = var_idvas;
```

```
END IF;
```

SELESAI ← 14



Gambar 5.13 Flow graph Pengujian Function generate_village_resource_schedule_from_resources_buildings

$$\begin{aligned}
 V(G) &= E - N + 2 \\
 &= 15 - 14 + 2 = 3
 \end{aligned}$$

Berdasarkan nilai *cyclomatic complexity* yang telah diperoleh dari hasil kalkulasi, maka bisa ditentukan dua buah basis set dari jalur *independent*, yakni :

Jalur 1 : 1 – 2 – 3 – 4 – 5 – 6 – 7 – 9 – 10 – 11 – 12 – 13 – 14
 Jalur 2 : 1 – 2 – 3 – 4 – 5 – 6 – 8 – 9 – 10 – 11 – 12 – 13 – 14
 Jalur 3 : 1 – 2 – 3 – 4 – 5 – 6 – 7 – 9 – 10 – 11 – 12 – 14

Penjelasan kasus uji pengujian *procedure generate_village_resource_schedule_from_resources_buildings* akan dipaparkan pada tabel 5.26

Tabel 5.26 Kasus Uji pada Pengujian *Procedure generate_village_resource_schedule_from_resources_buildings*

Jalur	Kasus Uji	Hasil yang diharapkan	Hasil yang didapatkan
1	Menjalankan <i>procedure</i> dengan data masukan jenis bangunan penghasil sumber daya tat kala tabel <i>villages_resources_schedules</i> masih kosong.	Pembangunan bangunan berdampak <i>generate</i> jadwal penghasilan sumber daya baru pada tabel <i>villages_resources_schedules</i> dengan nilai idvas sama dengan 1.	Pembangunan bangunan berdampak <i>generate</i> jadwal penghasilan sumber daya baru pada tabel <i>villages_resources_schedules</i> dengan nilai idvas sama dengan 1.
2	Menjalankan <i>procedure</i> dengan data masukan jenis bangunan	Pembangunan bangunan berdampak <i>generate</i> jadwal penghasilan sumber daya baru pada tabel <i>villages_resources_schedules</i>	Pembangunan bangunan berdampak <i>generate</i> jadwal penghasilan sumber daya baru pada tabel <i>villages_resources_schedules</i>

	penghasil sumber daya tatkala tabel villages_resources_schedules sudah berisi.	les dengan nilai idvas sama dengan nilai idvas terbesar dalam tabel ditambah dengan 1.	les dengan nilai idvas sama dengan nilai idvas terbesar dalam tabel ditambah dengan 1.
3	Menjalankan <i>procedure</i> dengan data masukan jenis bangunan non-penghasil sumber daya tatkala tabel villages_resources_schedules masih kosong..	Pembangunan bangunan tidak berdampak <i>generate</i> jadwal penghasilan sumber daya baru pada tabel villages_resources_schedules	Pembangunan bangunan tidak berdampak <i>generate</i> jadwal penghasilan sumber daya baru pada tabel villages_resources_schedules

6. Pengujian Integrasi pada Procedure order_attack

Tabel 5.27 Pemodelan *Flow Graph* Algoritma *Procedure order_attack*

Pseudocode Procedure order_attack	
Deklarasi awal function get_new_attack	
var_attack:INT	
Deskripsi	
MULAI ← 1	
SELECT MAX(attack) INTO var_attack FROM villages_troops_attacks LOCK IN SHARE MODE;	} 2
IF var attack IS NULL ← 3	

```

THEN
    SET var_attack = 0; ← 4
ELSE
    SET var_attack = var_attack; ← 5
END IF;

SET var_attack = var_attack + 1; } 6
RETURN var_attack;

SELESAI ← 7

Deklarasi awal function get_attribute_id

var_attribute_name:VARCHAR(45) -> Masukan dari user/sistem
var_ida:INT

Deskripsi function get_attribute_id

MULAI ← 8

SELECT ida INTO var_ida
FROM attributes
WHERE attribute = var_attribute_name; } 9

RETURN var_ida;

SELESAI ← 10

Deklarasi awal function get_village_axis

var_village:INT -> Masukan dari user/sistem
var_axis:VARCHAR(50) -> Masukan dari user/sistem
var_attribute:INT
var_value:INT

Deskripsi function get_village_axis

MULAI ← 11

SET var_attribute = get_attribute_id(var_axis); }
SELECT value INTO var_value
FROM villages_locations
WHERE village = var_village
AND attribute = var_attribute; } 12

```



```
RETURN var_value;
```

```
SELESAI ← 13
```

Deklarasi awal procedure get_village_location

```
var_village:INT -> Masukan dari user/sistem
```

```
var_x_axis:INT -> Keluaran dari sistem
```

```
var_y_axis:INT -> Keluaran dari sistem
```

Deskripsi procedure get_village_location

```
MULAI ← 14
```

```
SET var_x_axis =  
get_village_axis(var_village,"x_axis");
```

```
SET var_y_axis =  
get_village_axis(var_village,"y_axis");
```

15

```
SELESAI ← 16
```

Deklarasi awal function get_distance

```
var_village:INT -> Masukan dari user/sistem
```

```
var_target_village:INT -> Masukan dari user/sistem
```

```
var_village_x:INT
```

```
var_village_y:INT
```

```
var_target_village_x:INT
```

```
var_target_village_y:INT
```

```
var_distance:FLOAT
```

Deskripsi function get_distance

```
MULAI ← 17
```

```
CALL get_village_location  
(var_village,var_village_x,var_village_y);
```

```
CALL get_village_location  
(var_target_village,var_target_village_x,  
var_target_village_y);
```

18

```
SET var distance = SQRT(POW(var target village x-
```

```
var_village_x,2)+POW(var_target_village_y-  
var_village_y,2));
```

```
RETURN var_distance;
```

SELESAI ← 19

Deklarasi awal function get_travel_time

```
var_troop:INT -> Masukan dari user/sistem  
var_village:INT -> Masukan dari user/sistem  
var_target_village:INT -> Masukan dari user/sistem  
var_travel_time:TIME  
var_distance:INT  
var_speed:INT
```

Deskripsi function get_travel_time

MULAI ← 20

```
SET var_distance =  
get_distance(var_village,var_target_village);
```

```
SELECT iv.value INTO var_speed  
FROM troops_stats ts  
LEFT JOIN mem_integer_variables iv  
ON ts.value_iv = iv.idiv  
LEFT JOIN attributes a ON ts.attribute = a.ida  
WHERE a.attribute = "troop_speed"  
AND ts.troop = var_troop;  
SET var_travel_time := SEC_TO_TIME  
(var_distance / var_speed * 3600);
```

21

```
RETURN var_travel_time;
```

SELESAI ← 22

Deklarasi awal function get_troop_id

```
var_troop:VARCHAR(45) -> Masukan dari user/sistem  
var_idt:INT
```

Deskripsi function get_troop_id

MULAI ← 23

```

SELECT idt INTO var_idt
FROM troops
WHERE troop = var_idt;
RETURN var_idt;
SELESAI ← 24

Deklarasi awal function get_new_idvt
var_idvt:INT

Deskripsi function get_new_idvt
MULAI ← 26

SELECT MAX(idvt)
INTO var_idvt
FROM villages_troops LOCK IN SHARE MODE;
IF var_idvt IS NULL THEN
    SET var_idvt = 0;
ELSE
    SET var_idvt = var_idvt;
END IF;
SET var_idvt = var_idvt + 1;
RETURN var_idvt;
SELESAI ← 32

Deklarasi awal function get_varchar_variables_id
var_variable:VARCHAR(45) -> Masukan dari user/sistem
var_idvv:INT

Deskripsi function get_varchar_variables_id
MULAI ← 33

SELECT idvv INTO var_idvv
FROM mem_varchar_variables
WHERE variable = var_variable;
RETURN var idvv;

```


SELESAI ← 35

Deklarasi awal

```
var_village:INT -> Masukan dari user/sistem
var_troop:VARCHAR(50) -> Masukan dari user/sistem
var_target_village:INT -> Masukan dari user/sistem
var_process:VARCHAR(50)
var_next_process:VARCHAR(50)
var_marker_a:INT
var_troop_type:VARCHAR(50)
var_troop_id:INT
var_troop_amount:INT
var_marker_k:INT
var_attack:INT
var_idvt:INT
var_status_away:INT
var_verificator:INT
var_speedless_troop:INT
var_speed_value:INT
var_attribute_id:INT
var_travel_time:TIME
var_status_id:INT
```

Deskripsi

MULAI ← 36

SET var_attack = get_new_attack(); ← 37

REPEAT ← 38

SET var_marker_a = LOCATE("&",var_troop); ← 39

IF var_marker_a != 0 THEN ← 40

```
    SET var_process = SUBSTRING(var_troop,1,
    var_marker_a - 1);
    SET var_next_process = SUBSTRING(var_troop,
    var_marker_a + 1);
```

41

ELSE

```
    SET var_process = SUBSTRING(var_troop,
    var marker a + 1);
```

42

```

END IF;

SET var_troop = var_next_process;

SET var_marker_k = LOCATE("#",var_process);

SET var_troop_type = SUBSTRING(var_process,1,
var_marker_k - 1);

SET var_troop_amount = SUBSTRING(var_process,
var_marker_k + 1);

SET var_troop_id = get_troop_id(var_troop_type);

SELECT IFNULL(SUM(value),0) INTO
var_verificator FROM villages_troops
WHERE village = var_village AND troop = var_troop_id;

IF var_troop_amount > var_verificator THEN ← 43

    SET SQL_SAFE_UPDATES=0;

    DELETE villages_troops,villages_troops_attacks
    FROM villages_troops
    INNER JOIN villages_troops_attacks
    ON villages_troops.idvt = villages_troops_
    attacks.village_troop
    WHERE villages_troops_attacks.attack
    = var_attack;
    SET SQL_SAFE_UPDATES=1;

    SET var_marker_a = 0;

ELSE

    IF var_troop_amount != 0 THEN ← 46

        SET var_idvt = get_new_idvt();

        SET var_status_away =
        get_varchar_variables_id("troop_away");

        INSERT INTO villages_troops
        SET
        idvt = var_idvt,
        village = var_village,

```

43

45

47

```

        troop = var_troop_id,
        value = var_troop_amount * -1,
        execution_time = NOW(),
        status_vv = var_status_away;

        INSERT INTO villages_troops_attacks
        SET
        attack = var_attack,
        village_troop = var_idvt;

    END IF;

END IF;

UNTIL var_marker_a = 0 END REPEAT; ← 48

SET var_attribute_id = get_attribute_id
("troop_speed");

SELECT vt.troop, MIN(iv.value) INTO
var_speedless_troop, var_speed_value
FROM villages_troops_attacks vta
LEFT JOIN villages_troops vt ON vt.idvt =
vta.village_troop
LEFT JOIN troops_stats ts ON ts.troop = vt.troop
LEFT JOIN mem_integer_variables iv ON iv.idiv =
ts.value_iv
WHERE ts.attribute = var_attribute_id
AND vta.attack = var_attack;

IF var_speedless_troop IS NOT NULL THEN ← 50

    SET var_travel_time =
    get_travel_time(var_speedless_troop,
    var_village, var_target_village);

    SET var_status_id =
    get_varchar_variables_id("troop_away");

    INSERT INTO villages_troops_attacks_schedules
    SET
    village_troop_attack = var_attack,
    target_village = var_target_village,
    execution_time = ADDTIME

```

49

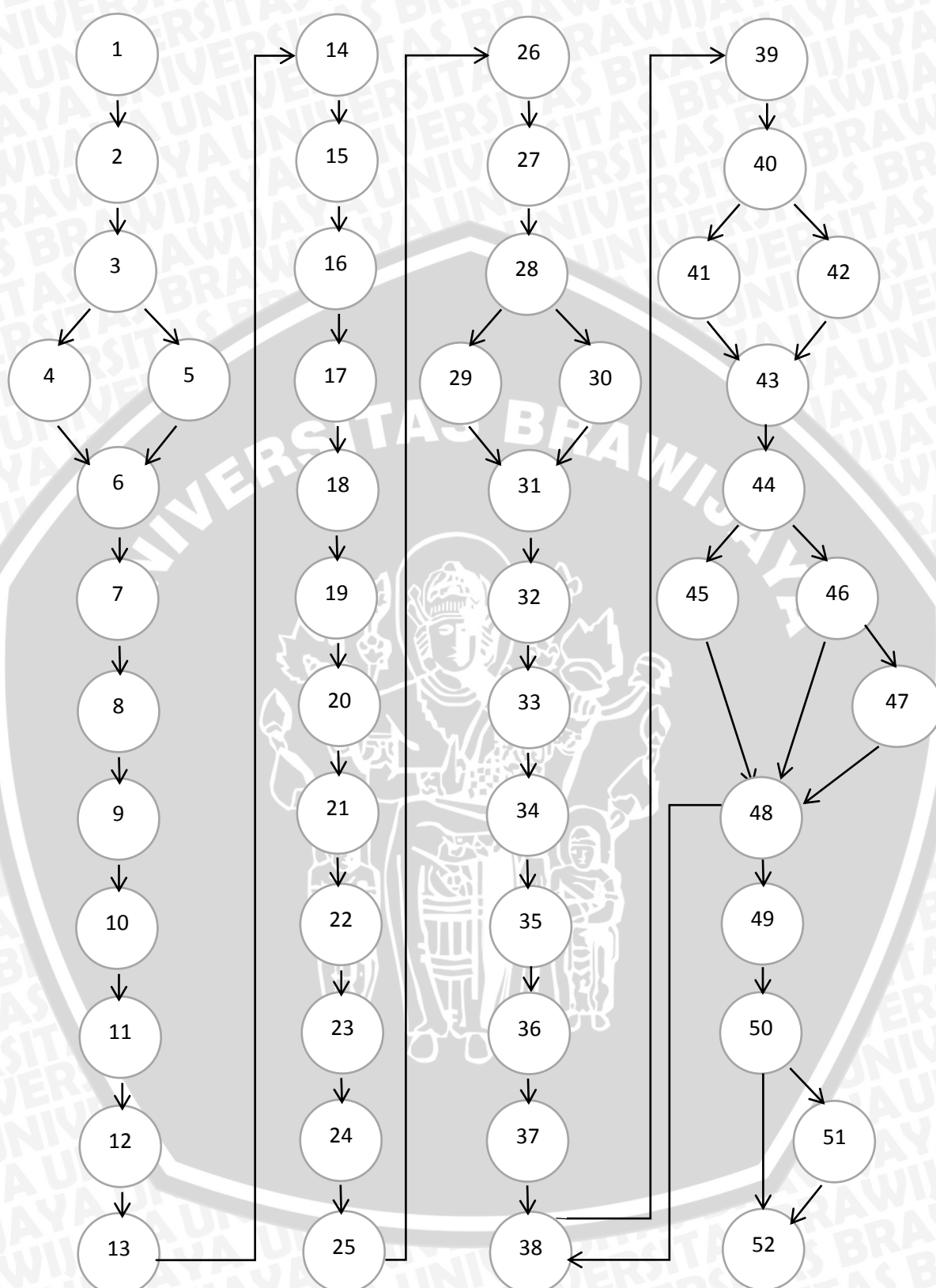
51


```
(NOW(),var_travel_time),  
status_vv = var_status_id;
```

```
END IF;
```

```
SELESAI ← 52
```





Gambar 5.14 Flow graph Pengujian Procedure order_attack

$$\begin{aligned} V(G) &= E - N + 2 \\ &= 58 - 52 + 2 = 8 \end{aligned}$$

Berdasarkan nilai *cyclomatic complexity* yang telah diperoleh dari hasil kalkulasi, maka bisa ditentukan delapan buah basis set dari jalur *independent*, yakni :

- Jalur 1 : 1 – 2 – 3 – 4 – 6 – 7 – 8 – 9 – 10 – 11 – 12 – 13 – 14 – 15
– 16 – 17 – 18 – 19 – 20 – 21 – 22 – 23 – 24 – 25 – 26 –
27 – 28 – 30 – 31 – 32 – 33 – 34 – 35 – 36 – 37 – 38 – 39 – 40
– 41 – 43 – 44 – 46 – 47 – 48 – 49 – 50 – 51 – 52
- Jalur 2 : 1 – 2 – 3 – 4 – 6 – 7 – 8 – 9 – 10 – 11 – 12 – 13 – 14 – 15
– 16 – 17 – 18 – 19 – 20 – 21 – 22 – 23 – 24 – 25 – 26 –
27 – 28 – 29 – 31 – 32 – 33 – 34 – 35 – 36 – 37 – 38 – 39 – 40
– 41 – 43 – 44 – 46 – 47 – 48 – 49 – 50 – 51 – 52
- Jalur 3 : 1 – 2 – 3 – 5 – 6 – 7 – 8 – 9 – 10 – 11 – 12 – 13 – 14 – 15
– 16 – 17 – 18 – 19 – 20 – 21 – 22 – 23 – 24 – 25 – 26 –
27 – 28 – 29 – 31 – 32 – 33 – 34 – 35 – 36 – 37 – 38 – 39 – 40
– 41 – 43 – 44 – 46 – 47 – 48 – 49 – 50 – 51 – 52
- Jalur 4 : 1 – 2 – 3 – 5 – 6 – 7 – 8 – 9 – 10 – 11 – 12 – 13 – 14 – 15
– 16 – 17 – 18 – 19 – 20 – 21 – 22 – 23 – 24 – 25 – 26 –
27 – 28 – 30 – 31 – 32 – 33 – 34 – 35 – 36 – 37 – 38 – 39 – 40
– 41 – 43 – 44 – 46 – 47 – 48 – 49 – 50 – 51 – 52
- Jalur 5 : 1 – 2 – 3 – 4 – 6 – 7 – 8 – 9 – 10 – 11 – 12 – 13 – 14 – 15
– 16 – 17 – 18 – 19 – 20 – 21 – 22 – 23 – 24 – 25 – 26 –
27 – 28 – 30 – 31 – 32 – 33 – 34 – 35 – 36 – 37 – 38 – 39 – 40
– 42 – 43 – 44 – 46 – 47 – 48 – 49 – 50 – 51 – 52
- Jalur 6 : 1 – 2 – 3 – 4 – 6 – 7 – 8 – 9 – 10 – 11 – 12 – 13 – 14 – 15
– 16 – 17 – 18 – 19 – 20 – 21 – 22 – 23 – 24 – 25 – 26 –

27 - 28 - 29 - 31 - 32 - 33 - 34 - 35 - 36 - 37 - 38 - 39 - 40
 - 42 - 43 - 44 - 46 - 47 - 48 - 49 - 50 - 51 - 52

Jalur 7 : 1 - 2 - 3 - 5 - 6 - 7 - 8 - 9 - 10 - 11 - 12 - 13 - 14 - 15
 - 16 - 17 - 18 - 19 - 20 - 21 - 22 - 23 - 24 - 25 - 26 -
 27 - 28 - 29 - 31 - 32 - 33 - 34 - 35 - 36 - 37 - 38 - 39 - 40
 - 42 - 43 - 44 - 46 - 47 - 48 - 49 - 50 - 51 - 52

Jalur 8 : 1 - 2 - 3 - 5 - 6 - 7 - 8 - 9 - 10 - 11 - 12 - 13 - 14 - 15
 - 16 - 17 - 18 - 19 - 20 - 21 - 22 - 23 - 24 - 25 - 26 -
 27 - 28 - 30 - 31 - 32 - 33 - 34 - 35 - 36 - 37 - 38 - 39 - 40
 - 42 - 43 - 44 - 46 - 47 - 48 - 49 - 50 - 51 - 52

Penjelasan kasus uji pengujian *procedure order_attack* akan dipaparkan pada tabel 5.28

Tabel 5.28 Kasus Uji pada Pengujian *Procedure order_attack*

Jalur	Kasus Uji	Hasil yang diharapkan	Hasil yang didapatkan
1	Menjalankan <i>procedure</i> ketika belum terdapat data sama sekali dalam tabel <i>villages_troops_attack</i> s dan <i>villages_troops</i> dengan masukan 1 jenis pasukan.	Masukan pada <i>procedure order_attack</i> tidak sampai proses terdaftar pada tabel penjadwalan penyerangan pasukan karena jumlah pasukan tidak memungkinkan.	Masukan pada <i>procedure order_attack</i> tidak sampai proses terdaftar pada tabel penjadwalan penyerangan pasukan karena jumlah pasukan tidak memungkinkan.
2	Menjalankan <i>procedure</i> dengan masukan yang benar ketika belum terdapat	Masukan pada <i>procedure order_attack</i> berupa serangan pemain terhadap suatu desa	Masukan pada <i>procedure order_attack</i> berupa serangan pemain terhadap suatu

	data sama sekali dalam tabel villages_troops_attacks dengan masukan 1 jenis pasukan.	berhasil terdaftar pada tabel penjadwalan penyerangan pasukan.	desa berhasil terdaftar pada tabel penjadwalan penyerangan pasukan.
3	Menjalankan <i>procedure</i> ketika belum terdapat data sama sekali dalam tabel dan villages_troops dengan masukan 1 jenis pasukan.	Masukan pada <i>procedure order_attack</i> tidak sampai proses terdaftar pada tabel penjadwalan penyerangan pasukan karena jumlah pasukan tidak memungkinkan.	Masukan pada <i>procedure order_attack</i> tidak sampai proses terdaftar pada tabel penjadwalan penyerangan pasukan karena jumlah pasukan tidak memungkinkan.
4	Menjalankan <i>procedure</i> dengan masukan yang benar ketika sudah terdapat data dalam tabel villages_troops_attacks dengan masukan 1 jenis pasukan.	Masukan pada <i>procedure order_attack</i> berupa serangan pemain terhadap suatu desa berhasil terdaftar pada tabel penjadwalan penyerangan pasukan.	Masukan pada <i>procedure order_attack</i> berupa serangan pemain terhadap suatu desa berhasil terdaftar pada tabel penjadwalan penyerangan pasukan.
5	Menjalankan <i>procedure</i> ketika belum terdapat data sama sekali dalam tabel villages_troops_attack s dan villages_troops	Masukan pada <i>procedure order_attack</i> tidak sampai proses terdaftar pada tabel penjadwalan penyerangan pasukan karena jumlah pasukan	Masukan pada <i>procedure order_attack</i> tidak sampai proses terdaftar pada tabel penjadwalan penyerangan pasukan karena jumlah pasukan

	dengan masukan lebih dari 1 jenis pasukan.	tidak memungkinkan.	tidak memungkinkan.
6	Menjalankan <i>procedure</i> dengan masukan yang benar ketika belum terdapat data sama sekali dalam tabel <i>villages_troops_attacks</i> dengan masukan lebih dari 1 jenis pasukan.	Masukan pada <i>procedure order_attack</i> berupa serangan pemain terhadap suatu desa berhasil terdaftar pada tabel penjadwalan penyerangan pasukan.	Masukan pada <i>procedure order_attack</i> berupa serangan pemain terhadap suatu desa berhasil terdaftar pada tabel penjadwalan penyerangan pasukan.
7	Menjalankan <i>procedure</i> ketika belum terdapat data sama sekali dalam tabel dan <i>villages_troops</i> dengan masukan lebih dari 1 jenis pasukan.	Masukan pada <i>procedure order_attack</i> tidak sampai proses terdaftar pada tabel penjadwalan penyerangan pasukan karena jumlah pasukan tidak memungkinkan.	Masukan pada <i>procedure order_attack</i> tidak sampai proses terdaftar pada tabel penjadwalan penyerangan pasukan karena jumlah pasukan tidak memungkinkan.
8	Menjalankan <i>procedure</i> dengan masukan yang benar ketika sudah terdapat data dalam tabel <i>villages_troops_attacks</i> dengan masukan lebih dari 1 jenis pasukan.	Masukan pada <i>procedure order_attack</i> berupa serangan pemain terhadap suatu desa berhasil terdaftar pada tabel penjadwalan penyerangan pasukan.	Masukan pada <i>procedure order_attack</i> berupa serangan pemain terhadap suatu desa berhasil terdaftar pada tabel penjadwalan penyerangan pasukan.

5.2.2 Analisis Pengujian Integrasi

Hasil analisis yang bisa diambil dari serangkaian tahapan pengujian integrasi yang telah dilakukan yakni:

1. Berdasarkan adanya kesesuaian antara seluruh hasil pengujian integrasi pada *function* atau *procedure* dengan *output* atau hasil keluaran yang diharapkan pada *game server online* permainan *strategy* Kerajaan Majapahit maka bisa ditarik kesimpulan bahwa integrasi dari unit *function* atau *procedure* pada perangkat lunak yang telah dikembangkan sudah mampu beroperasi sebagaimana mestinya serta mampu menghasilkan *output* sesuai dengan yang diharapkan.
2. Berdasarkan hasil kalkulasi atau perhitungan *cyclomatic complexity* dari tiap *flow graph* algoritma *function* atau *procedure* yang telah diuji, Jumlah jalur kasus uji bisa bermacam-macam bergantung dari kompleksitas algoritma *function* yang mengacu pada banyaknya seleksi kondisi. Hasil *cyclomatic complexity* terbanyak yang ditemukan saat proses pengujian integrasi terdapat pada *procedure order_attack*.

5.3 Pengujian Validasi

Pengujian validasi dilakukan untuk mengetahui sejauh mana kinerja sistem yang telah dibangun, apakah telah mampu beroperasi secara benar, sebagaimana mestinya serta sesuai dengan spesifikasi yang telah dibuat. Daftar kebutuhan fungsional dan non-fungsional akan menjadi pedoman atau acuan untuk melaksanakan pengujian validasi. Pengujian validasi menggunakan metode pengujian *Black Box* karena tidak memerlukan perhatian khusus atau konsentrasi terhadap alur jalannya algoritma perangkat lunak.

5.3.1 Daftar kebutuhan fungsional dan non-fungsional

Daftar lengkap berbagai macam kebutuhan fungsional serta nonfungsional yang difungsikan untuk memberi penjelasan terhadap seluruh ragam fitur permainan yang mengacu secara penuh pada *use case* perangkat lunak aplikasi

game server online permainan *strategy* Kerajaan Majapahit akan dipaparkan secara rinci pada tabel 5.29.

Tabel 5.29 Daftar Berbagai Macam Kebutuhan Fungsional dan Non-Fungsional dari *Game Server Online* Permainan *Strategy* Kerajaan Majapahit

ID	Requirements	Aktor	Nama Use Case
F01	Perangkat lunak permainan harus menyediakan proses pendaftaran calon pemain ke <i>game server online</i> .	Game Client	sign_up
F02	Perangkat lunak permainan harus menyediakan keluaran berupa ID dari desa pertama berdasarkan masukan <i>username</i> pemain.	Game Client	default_active_village
F03	Perangkat lunak permainan harus menyediakan proses verifikasi dari masukan <i>username</i> serta <i>password</i> pemain dan kemudian menyediakan keluaran berupa hasil status verifikasi.	Game Client	log_in
F04	Perangkat lunak permainan harus menyediakan keluaran berupa peringkat dari desa pemain di <i>game server online</i> berdasarkan masukan berupa ID desa pemain.	Game Client	village_ranking
F05	Perangkat lunak permainan harus menyediakan keluaran berupa daftar peringkat seluruh pemain pada <i>game server online</i>	Game Client	villages_rankings
F06	Perangkat lunak permainan harus menyediakan keluaran berupa	Game Client	village_location

	koordinat lokasi dari suatu desa lengkap dengan data waktu tempuh pasukan dari desa pemain tatkala dilakukan penyerangan berdasarkan masukan berupa ID desa pemain dan nama desa target serangan.		
F07	Perangkat lunak permainan harus menyediakan keluaran berupa koordinat lokasi dari seluruh desa dalam <i>game server online</i> lengkap dengan data waktu tempuh pasukan dari desa pemain tatkala dilakukan penyerangan berdasarkan masukan berupa ID desa pemain.	Game Client	villages_locations
F08	Perangkat lunak permainan harus menyediakan keluaran berupa data lengkap sumber daya desa berdasarkan masukan berupa ID desa pemain.	Game Client	village_resources
F09	Perangkat lunak permainan harus menyediakan keluaran berupa daftar sumber daya yang diperlukan untuk membangun segala jenis bangunan yang dapat dibangun dalam permainan <i>strategy</i> Kerajaan Majapahit.	Game Client	building_prices
F10	Perangkat lunak permainan harus menyediakan proses untuk melakukan pembangunan bangunan di desa pemain berdasarkan masukan	Game Client	order_building

	berupa ID desa pemain dan nama bangunan yang ingin dibangun.		
F11	Perangkat lunak permainan harus menyediakan keluaran berupa jumlah bangunan tertentu yang ada di desa pemain berdasarkan masukan berupa ID desa pemain dan nama bangunan.	Game Client	village_building_amo unts
F12	Perangkat lunak permainan harus menyediakan keluaran berupa daftar jadwal seluruh proses pembangunan bangunan yang sedang berlangsung di desa pemain berdasarkan masukan berupa ID desa pemain.	Game Client	village_building_build _schedules
F13	Perangkat lunak permainan harus menyediakan keluaran berupa jumlah dari daftar jadwal seluruh proses pembangunan bangunan yang sedang berlangsung di desa pemain berdasarkan masukan berupa ID desa pemain.	Game Client	count_village_buildin g_build_schedules
F14	Perangkat lunak permainan harus menyediakan keluaran berupa daftar sumber daya yang diperlukan untuk melatih segala jenis pasukan yang dapat dilatih dalam permainan <i>strategy</i> Kerajaan Majapahit.	Game Client	troop_prices
F15	Perangkat lunak permainan harus menyediakan keluaran berupa daftar jadwal serangan pasukan desa lain terhadap desa pemain berdasarkan	Game Client	enemy_attack_schedul es

	masukannya berupa ID desa pemain.		
F16	Perangkat lunak permainan harus menyediakan keluaran berupa daftar jadwal serangan pasukan desa pemain terhadap desa lain berdasarkan masukan berupa ID desa pemain.	Game Client	troop_attack_schedules
F17	Perangkat lunak permainan harus menyediakan keluaran berupa daftar jadwal kembalinya pasukan desa pemain yang selamat dari pertempuran dengan desa lain berdasarkan masukan berupa ID desa pemain.	Game Client	troop_back_schedules
F18	Perangkat lunak permainan harus menyediakan keluaran berupa data lengkap jumlah pasukan yang dimiliki serta sedang berada di desa pemain berdasarkan masukan berupa ID desa pemain.	Game Client	village_troops
F19	Perangkat lunak permainan harus menyediakan keluaran berupa daftar data lengkap atribut dari seluruh pasukan dalam permainan.	Game Client	troop_stats
F20	Perangkat lunak permainan harus menyediakan proses untuk mengirimkan pasukan penyerang dari sebuah desa menuju ke desa lain berdasarkan masukan ID desa pemain, nama dan jumlah pasukan	Game Client	order_attack

	serta ID desa target serangan.		
F21	Perangkat lunak permainan harus menyediakan proses untuk melakukan pelatihan pasukan di desa berdasarkan masukan ID desa pemain dan jenis pasukan yang ingin dilatih.	Game Client	order_troop
F22	Perangkat lunak permainan harus menyediakan keluaran berupa jumlah jadwal keseluruhan serangan pasukan dari desa pemain ke desa lain yang sedang berlangsung berdasarkan masukan berupa ID desa pemain.	Game Client	count_troop_attack_schedules
F23	Perangkat lunak permainan harus menyediakan keluaran berupa jumlah jadwal keseluruhan kembalinya pasukan desa pemain dari desa lain yang sedang berlangsung berdasarkan masukan berupa ID desa pemain.	Game Client	count_troop_back_schedules
F24	Perangkat lunak permainan harus menyediakan keluaran berupa jumlah jadwal keseluruhan serangan pasukan dari desa lain ke desa pemain yang sedang berlangsung berdasarkan masukan berupa ID desa pemain.	Game Client	count_enemy_attack_schedules
F25	Perangkat lunak permainan harus menyediakan keluaran berupa jadwal	Game Client	village_troop_build_schedules

	keseluruhan pelatihan pasukan yang sedang berlangsung di desa pemain berdasarkan masukan berupa ID desa pemain.		
F26	Perangkat lunak permainan harus menyediakan berupa jumlah jadwal keseluruhan pelatihan pasukan yang sedang berlangsung di desa pemain berdasarkan masukan berupa ID desa pemain.	Game Client	count_village_troop_build_schedules
F27	Perangkat lunak permainan harus menyediakan keluaran berupa jadwal penghasilan sumber daya tertentu oleh bangunan penghasil sumber daya berdasarkan masukan berupa ID desa pemain dan jenis sumber daya.	Game Client	village_resource_schedules
F28	Perangkat lunak permainan harus menyediakan keluaran berupa jadwal datangnya sumber daya rampasan perang ke desa pemain berdasarkan masukan berupa ID desa pemain.	Game Client	war_loot_schedules
F29	Perangkat lunak permainan harus menyediakan keluaran berupa jumlah jadwal datangnya sumber daya rampasan perang ke desa pemain berdasarkan masukan berupa ID desa pemain.	Game Client	count_war_loot_schedules
N01	Perangkat lunak permainan harus menyediakan proses verifikasi	Game Client	username_availability

ketersediaan <i>username</i> yang dipilih calon pemain di <i>game server online</i> permainan <i>strategy</i> Kerajaan Majapahit.		

5.3.2 Kasus uji validasi

Keseluruhan kasus uji validasi pada tahapan pengujian validasi dalam pengembangan *game server online* permainan *strategy* Kerajaan Majapahit dijelaskan pada tabel 5.30.

Tabel 5.30 Kasus Uji Validasi pada Pengujian Validasi *Game Server Online* Permainan *Strategy* Kerajaan Majapahit

ID	Kasus Uji	Objek Uji	Prosedur dan Input	Kondisi yang Diharapkan
C01	Mendaftarkan pemain di game server online.	F01	Melakukan pendaftaran menggunakan <i>compatible game client</i> dengan masukan <i>username</i> yang belum terdaftar sebelumnya.	Calon pemain sukses mendaftar diri di <i>game server online</i> .
C02			Melakukan pendaftaran menggunakan <i>compatible game client</i> dengan masukan <i>username</i> yang sudah terdaftar	Calon pemain gagal mendaftar diri di <i>game server online</i> .

			sebelumnya..	
C03	Menampilkan ID desa pemain.	F02	Melakukan proses masuk ke dalam dunia permainan menggunakan <i>compatible game client</i> dengan masukan <i>username</i> dan <i>password</i> pemain.	<i>Game server online</i> memberikan keluaran berupa ID desa pemain dalam format JSON.
C04	Melakukan verifikasi <i>username</i> dan <i>password</i> pemain untuk masuk dunia permainan.	F03	Melakukan proses masuk ke dalam dunia permainan menggunakan <i>compatible game client</i> dengan masukan <i>username</i> dan <i>password</i> pemain yang benar.	<i>Game server online</i> memberikan keluaran berupa status verifikasi sukses dalam format JSON.
C05			Melakukan proses masuk ke dalam dunia permainan menggunakan <i>compatible game client</i> dengan masukan <i>username</i> dan <i>password</i> pemain yang salah.	<i>Game server online</i> memberikan keluaran berupa status verifikasi gagal dalam format JSON.
C06	Menampilkan peringkat desa	F04	Masuk ke dalam dunia permainan	<i>Game server online</i> memberikan keluaran

	pemain.		menggunakan <i>compatible game client</i> kemudian melihat peringkat dari desa pemain.	berupa peringkat desa pemain dalam format JSON.
C07	Menampilkan seluruh daftar peringkat desa pemain.	F05	Memilih menu untuk menampilkan daftar seluruh peringkat desa pada permainan menggunakan <i>compatible game client</i> .	<i>Game server online</i> memberikan keluaran berupa daftar peringkat seluruh desa pemain dalam format JSON.
C08	Menampilkan koordinat lokasi serta waktu tempuh pasukan desa pemain terhadap desa tujuan serangan.	F06	Memilih desa tujuan serangan pasukan pemain menggunakan <i>compatible game client</i> .	<i>Game server online</i> memberikan keluaran berupa koordinat lokasi serta waktu tempuh pasukan desa pemain terhadap desa tujuan serangan dalam format JSON.
C09	Menampilkan daftar koordinat lokasi dari seluruh desa pada <i>game server online</i> serta waktu tempuh pasukan desa pemain terhadap desa yang ditampilkan.	F07	Memilih menu untuk menampilkan daftar seluruh koordinat lokasi desa pada permainan menggunakan <i>compatible game client</i> .	<i>Game server online</i> memberikan keluaran berupa daftar koordinat lokasi dari seluruh desa pada <i>game server online</i> serta waktu tempuh pasukan desa pemain terhadap desa dalam format JSON.

C10	Menampilkan jumlah sumber daya tertentu yang dimiliki desa pemain.	F08	Masuk ke dalam dunia permainan menggunakan <i>compatible game client</i> kemudian melihat jumlah sumber daya di desa pemain.	<i>Game server online</i> memberikan keluaran berupa jumlah sumber daya desa tertentu dalam format JSON.
C11	Menampilkan daftar bangunan dalam permainan lengkap dengan sumber daya yang diperlukan untuk membangunnya.	F09	Memilih menu untuk menampilkan daftar bangunan pada permainan menggunakan <i>compatible game client</i> .	<i>Game server online</i> memberikan keluaran berupa daftar bangunan dalam permainan lengkap dengan sumber daya yang diperlukan untuk membangunnya dalam format JSON.
C12	Melakukan pembangunan bangunan.	F10	Melakukan pembangunan dengan masukan jenis bangunan yang memerlukan sumber daya lebih dari sumber daya yang dimiliki desa pemain ketika slot pembangunan tersedia.	Proses pembangunan tidak berlangsung atau tidak terdaftar di jadwal pembangunan bangunan.
C13			Melakukan pembangunan	Proses pembangunan berlangsung atau

			dengan masukan jenis bangunan yang bisa dibangun dengan sumber daya yang dimiliki desa pemain ketika slot pembangunan tersedia..	terdaftar di jadwal pembangunan bangunan.
C14			Melakukan pembangunan dengan masukan jenis bangunan yang bisa dibangun dengan sumber daya yang dimiliki desa pemain ketika slot pembangunan tidak tersedia.	Proses pembangunan tidak berlangsung atau tidak terdaftar di jadwal pembangunan bangunan.
C15			Melakukan pembangunan dengan masukan jenis bangunan yang memerlukan sumber daya lebih dari sumber daya yang dimiliki desa pemain ketika slot pembangunan tidak tersedia.	Proses pembangunan tidak berlangsung atau tidak terdaftar di jadwal pembangunan bangunan.
C16	Menampilkan	F11	Masuk dunia	<i>Game server online</i>

	jumlah bangunan tertentu yang dimiliki desa pemain.		permainan kemudian melihat jumlah bangunan tertentu menggunakan <i>compatible game client</i> .	memberikan keluaran berupa jumlah bangunan tertentu yang dimiliki desa pemain dalam format JSON.
C17	Menampilkan jadwal pembangunan bangunan yang sedang berlangsung desa pemain.	F12	Memilih menu untuk menampilkan jadwal pembangunan bangunan menggunakan <i>compatible game client</i> .	<i>Game server online</i> memberikan keluaran berupa jadwal lengkap pembangunan bangunan di desa pemain dalam format JSON.
C18	Menampilkan jumlah jadwal pembangunan bangunan yang sedang berlangsung desa pemain.	F13	Memilih menu untuk menampilkan jadwal pembangunan bangunan kemudian melihat jumlah jadwal pembangunan menggunakan <i>compatible game client</i> .	<i>Game server online</i> memberikan keluaran berupa jumlah jadwal pembangunan bangunan di desa pemain dalam format JSON.
C19	Menampilkan daftar pasukan dalam permainan lengkap dengan sumber daya yang diperlukan untuk melatihnya.	F14	Memilih menu untuk menampilkan daftar pasukan pada permainan menggunakan <i>compatible game client</i> .	<i>Game server online</i> memberikan keluaran berupa daftar pasukan dalam permainan lengkap dengan sumber daya yang diperlukan untuk

				melatihnya dalam format JSON.
C20	Menampilkan jadwal serangan pasukan dari desa lain ke desa pemain.	F15	Memilih menu untuk menampilkan jadwal serangan pasukan dari desa lain ke desa pemain menggunakan <i>compatible game client</i> .	<i>Game server online</i> memberikan keluaran berupa jadwal serangan pasukan dari desa lain ke desa pemain dalam format JSON.
C21	Menampilkan jadwal serangan pasukan desa pemain ke desa lain.	F16	Memilih menu untuk menampilkan jadwal serangan pasukan desa pemain ke desa lain menggunakan <i>compatible game client</i> .	<i>Game server online</i> memberikan keluaran berupa jadwal serangan pasukan desa pemain ke desa lain dalam format JSON.
C22	Menampilkan jadwal kembalinya pasukan desa pemain dari desa lain.	F17	Memilih menu untuk menampilkan jadwal kembalinya pasukan desa pemain dari desa lain menggunakan <i>compatible game client</i> .	<i>Game server online</i> memberikan keluaran berupa jadwal kembalinya pasukan desa pemain dari desa lain dalam format JSON.
C23	Menampilkan daftar keseluruhan pasukan pemain yang sedang berada di desa pemain.	F18	Memilih menu untuk daftar keseluruhan pasukan pemain yang sedang berada di desa pemain	<i>Game server online</i> memberikan keluaran berupa daftar keseluruhan pasukan pemain yang sedang

			menggunakan <i>compatible game client</i> .	berada di desa pemain dalam format JSON.
C24	Menampilkan daftar pasukan dalam permainan lengkap dengan data atribut yang dimilikinya.	F19	Memilih menu untuk menampilkan daftar pasukan pada permainan menggunakan <i>compatible game client</i> .	<i>Game server online</i> memberikan keluaran berupa daftar pasukan dalam permainan lengkap dengan data atribut yang dimilikinya dalam format JSON.
C25	Melakukan proses pengiriman pasukan untuk menyerang desa lain.	F20	Melakukan pengiriman pasukan untuk menyerang desa lain dengan masukan jumlah pasukan yang melebihi jumlah pasukan pada desa pemain.	Proses penyerangan tidak berlangsung atau tidak terdaftar di jadwal penyerangan pasukan pemain.
C26			Melakukan pengiriman pasukan untuk menyerang desa lain dengan masukan jumlah pasukan kurang dari sama dengan jumlah pasukan pada desa pemain.	Proses penyerangan berlangsung atau terdaftar di jadwal penyerangan pasukan pemain.
C27	Melakukan	F21	Melakukan pelatihan	Proses pelatihan

	pelatihan pasukan di dalam desa pemain.		pasukan dengan masukan jenis pasukan yang memerlukan sumber daya lebih dari sumber daya yang dimiliki desa pemain ketika slot pelatihan tersedia.	pasukan tidak berlangsung atau tidak terdaftar di jadwal pelatihan pasukan.
C28			Melakukan pelatihan pasukan dengan masukan jenis pasukan yang bisa dibangun dengan sumber daya yang dimiliki desa pemain ketika slot pelatihan tersedia.	Proses pelatihan pasukan berlangsung atau terdaftar di jadwal pelatihan.
C29			Melakukan pelatihan pasukan dengan masukan jenis pasukan yang bisa dibangun dengan sumber daya yang dimiliki desa pemain ketika slot pelatihan tidak tersedia.	Proses pelatihan tidak berlangsung atau tidak terdaftar di jadwal pelatihan pasukan.
C30			Melakukan pelatihan pasukan dengan masukan jenis	Proses pelatihan tidak berlangsung atau tidak terdaftar di

			pasukan yang memerlukan sumber daya lebih dari sumber daya yang dimiliki desa pemain ketika slot pelatihan tidak tersedia.	jadwal pelatihan pasukan.
C31	Menampilkan jumlah jadwal serangan pasukan desa pemain ke desa lain.	F22	Memilih menu untuk menampilkan jadwal serangan pasukan desa pemain ke desa lain kemudian melihat jumlah jadwalnya menggunakan <i>compatible game client</i> .	<i>Game server online</i> memberikan keluaran berupa jumlah jadwal serangan pasukan desa pemain ke desa lain dalam format JSON.
C32	Menampilkan jumlah jadwal kembalinya pasukan desa pemain dari desa lain.	F23	Memilih menu untuk menampilkan jadwal kembalinya desa pemain dari desa lain kemudian melihat jumlah jadwalnya menggunakan <i>compatible game client</i> .	<i>Game server online</i> memberikan keluaran berupa jumlah jadwal kembalinya pasukan desa pemain dari desa lain dalam format JSON.
C33	Menampilkan jumlah jadwal serangan pasukan desa lain ke desa	F24	Memilih menu untuk menampilkan jadwal serangan pasukan desa lain ke desa	<i>Game server online</i> memberikan keluaran berupa jumlah jadwal serangan pasukan

	pemain.		pemain kemudian melihat jumlah jadwalnya menggunakan <i>compatible game client</i> .	desa lain ke desa pemain dalam format JSON.
C34	Menampilkan daftar jadwal pelatihan pasukan di desa pemain.	F25	Memilih menu untuk menampilkan daftar jadwal pelatihan pasukan di desa pemain menggunakan <i>compatible game client</i> .	<i>Game server online</i> memberikan keluaran berupa daftar jadwal pelatihan pasukan di desa pemain dalam format JSON.
C35	Menampilkan jumlah jadwal pelatihan pasukan di desa pemain.	F26	Memilih menu untuk menampilkan daftar jadwal pelatihan pasukan di desa pemain kemudian melihat jumlah jadwalnya menggunakan <i>compatible game client</i> .	<i>Game server online</i> memberikan keluaran berupa jumlah jadwal pelatihan pasukan di desa pemain dalam format JSON.
C36	Menampilkan daftar jadwal penghasilan sumber daya oleh bangunan penghasil sumber	F27	Memilih menu untuk menampilkan daftar jadwal penghasilan sumber daya oleh bangunan penghasil sumber daya di desa	<i>Game server online</i> memberikan keluaran berupa daftar jadwal penghasilan sumber daya oleh bangunan penghasil sumber

	daya di desa pemain.		pemain menggunakan <i>compatible game client</i> .	daya di desa pemain dalam format JSON.
C37	Menampilkan daftar jadwal datangnya sumber daya rampasan perang ke desa pemain.	F28	Memilih menu untuk menampilkan daftar jadwal datangnya sumber daya rampasan perang ke desa pemain menggunakan <i>compatible game client</i> .	<i>Game server online</i> memberikan keluaran berupa daftar jadwal datangnya sumber daya rampasan perang ke desa pemain dalam format JSON.
C38	Menampilkan jumlah jadwal datangnya sumber daya rampasan perang ke desa pemain.	F29	Memilih menu untuk menampilkan daftar jadwal datangnya sumber daya rampasan perang ke desa pemain kemudian melihat jumlah jadwalnya menggunakan <i>compatible game client</i> .	<i>Game server online</i> memberikan keluaran berupa jumlah jadwal datangnya sumber daya rampasan perang ke desa pemain dalam format JSON.
C39	Melakukan verifikasi ketersediaan <i>username</i> yang dipilih calon pemain pada <i>game</i>	N01	Pada saat proses pendaftaran dilakukan verifikasi ketersediaan <i>username</i> yang dipilih calon pemain	<i>Game server online</i> memberikan keluaran berupa status hasil verifikasi ketersediaan <i>username</i> dalam format JSON.

	<i>server online.</i>		<i>pada game server online. menggunakan compatible game client.</i>	

5.3.3 Hasil Pengujian Validasi

Hasil pengujian validasi yang dilaksanakan pada tahap pengujian validasi dalam pengembangan game server online permainan strategy Kerajaan Majapahit dijelaskan pada tabel 5.31.

Tabel 5.31 Hasil Uji Validasi pada Pengujian Validasi Game Server Online Permainan Strategy Kerajaan Majapahit

No	Kasus Uji	Objek Uji	Hasil
1	C01	F01	Valid
2	C02		Valid
3	C03	F02	Valid
4	C04	F03	Valid
5	C05		Valid
6	C06	F04	Valid
7	C07	F05	Valid
8	C08	F06	Valid
9	C09	F07	Valid
10	C10	F08	Valid
11	C11	F09	Valid
12	C12	F10	Valid
13	C13		Valid
14	C14		Valid
15	C15	F11	Valid
16	C16		Valid
17	C17	F12	Valid

18	C18	F13	Valid
19	C19	F14	Valid
20	C20	F15	Valid
21	C21	F16	Valid
22	C22	F17	Valid
23	C23	F18	Valid
24	C24	F19	Valid
25	C25	F20	Valid
26	C26		Valid
27	C27	F21	Valid
28	C28		Valid
29	C29		Valid
30	C30		Valid
31	C31	F22	Valid
32	C32	F23	Valid
33	C33	F24	Valid
34	C34	F25	Valid
35	C35	F26	Valid
36	C36	F27	Valid
37	C37	F28	Valid
38	C38	F29	Valid
39	C39	N01	Valid

5.3.4 Analisis Pengujian Validasi

Berdasarkan pengujian validasi yang telah dilakukan, didapatkan kesesuaian antara seluruh hasil uji validasi terhadap kinerja serta fungsionalitas dari *game server online* dengan hasil yang diharapkan pada daftar kebutuhan fungsional dan non-fungsional dari permainan *strategy* Kerajaan Majapahit. Oleh karenanya, dapat disimpulkan bahwa dari segi fungsionalitas, *game server online* yang dikembangkan dalam proses implementasi permainan telah mampu bekerja

sebagaimana mestinya dalam memenuhi seluruh kebutuhan yang telah dijabarkan dalam daftar kebutuhan.



BAB VI

KESIMPULAN DAN SARAN

6.1 Kesimpulan

1. *Gameplay* dan desain perangkat lunak pada *game server online* permainan *strategy* Kerajaan Majapahit dirancang dengan menggunakan *game concept design* dan *technical design*.
2. Implementasi perangkat lunak pada *game server online* permainan *strategy* Kerajaan Majapahit dilakukan dengan implementasi algoritma *gameplay* yang meliputi pembuatan *function*, *procedure* dan *event* pada MySQL serta implementasi *web* yang meliputi pembuatan berbagai kelas *model*, *controller* dan *view* menggunakan *framework Code Igniter*.
3. Hasil pengujian pada perangkat lunak pada *game server online* permainan *strategy* Kerajaan Majapahit menunjukkan adanya kesesuaian antara seluruh hasil pengujian unit, pengujian integrasi maupun pengujian validasi dengan *output* atau hasil keluaran yang diharapkan sehingga bisa ditarik kesimpulan bahwa perangkat lunak yang dikembangkan telah mampu beroperasi sebagaimana mestinya dan mampu menghasilkan *output* sesuai dengan yang diharapkan.

6.2 Saran

1. Pengembangan sistem keamanan berupa penggunaan HASH pada *password* pemain yang nantinya diimplementasikan pada algoritma pendaftaran pemain dan algoritma masuk dunia permainan.
2. Penambahan objek permainan, bisa berupa penambahan jenis pasukan, penambahan jenis bangunan, penambahan jenis sumber daya serta penambahan jenis atribut.

3. Penambahan fitur permainan berupa fitur sarana komunikasi baik antara pemain dengan pemain lain maupun antara pemain dengan administrator, bisa berupa *chatbox* atau aplikasi *messaging* dengan *inbox* dan *send message* sebagai fungsi minimal, fitur perdagangan antar pemain, fitur aliansi, fitur *historic event* serta fitur sistem administrasi *game server online* untuk *administrator* agar *administrator* dapat memantau dunia permainan, memantau kinerja sistem, dan mengambil tindakan terhadap berbagai masalah dengan lebih cepat dan lebih mudah.



DAFTAR PUSTAKA

- [AN1-12] Anonymous. 2012. "Pengenalan MySQL". Akses dari : <http://www.swalt.info/web/mysql/88-pengenalan-mysql.html>.
Tanggal Akses : 05 Mei 2014.
- [AN2-13] Anonymous. 2013. "Pengenalan Code Igniter". Akses dari : <http://codeigniter.tutorial.maestro-media.com/modul1>.
Tanggal Akses : 05 Mei 2014.
- [AN3-09] Anonymous. 2009. "Diagram Entitas Relasional (Diagram E-R) Bag.I". Akses dari : http://ndoware.com/diagram_e-r-bag_i.html.
Tanggal Akses : 14 Juli 2014.
- [AN4-13] Anonymous. 2013. "Kominfo : Pengguna Internet di Indonesia 63 Juta Orang". Akses dari : http://kominfo.go.id/index.php/content/detail/3415/Kominfo+%3A+Pengguna+Internet+di+Indonesia+63+Juta+Orang/0/berita_satker#.U6MylbGrIJQ.
Tanggal Akses : 05 Mei 2014.
- [DHA-06] Dharwiyanti, Sri dan Satria, Romie W. 2003. "Pengantar Unified Modeling Language (UML)". IlmuKomputer.Com.
- [ESR-12] Entertainment Software Rating Board (official Page). 2012. "Game Ratings & Descriptor Guide". Akses dari http://www.esrb.org/ratings/ratings_guide.jsp#rating_symbols.
Tanggal akses : 05 Mei 2014.
- [PRE-10] Pressman, Roger S. 2010. "Software Engineering: A Practitioner's Approach", 7th edition. Mc Graw Hill.
- [REP-08] Pedersen, Roger E. 2008. "Game Design Foundations, 2nd Edition". Wordware Publishing, Inc.

- [ROS-10] Roger, Scott E. 2010. *“Level Up! The Guide to Great VideoGame Design”*. John Wiley & Sons, Ltd.
- [SAP-11] Windi, Puput., Fathoni, Kholid., Yuniar, Rizky. 2011. *Rancang Bangun Game RPG Sejarah Kerajaan-Kerajaan Di Indonesia Dengan Pembelajaran Bahasa Jawa Untuk Anak Anak Dan Remaja*.
- [SEP-13] Septian, Ahmad Nur. 2011. *Rancang Bangun Game Server Lineage II Freya High Five Menggunakan L2J Server*.
- [SRJ-12] Srinivas, Nidhra and Jagruthi, Dondeti. 2012. *“Black Box and White Box Testing Techniques – A Literature Review”*. International Journal of Embedded Systems and Applications (IJESA), Vol.2, No.2, hal 29-50.
- [WEN-12] Wennendy, Budi P., Angjasrin, Febriyanto., Marcellinus, Yonathan. 2012. *Rancang Bangun Aplikasi Game Edukasi Multiplayer Berbasis Platform Blackberry*.
- [WIJ-12] Wijaya, Ekaprana., Sari, Yunita K., Kartikadarma, Etika. 2012. *Game Kebudayaan Sebagai Salah Satu Bentuk Pelestarian Kebudayaan dan Media Pembelajaran*.



