RANCANG BANGUN *MOBILE GAME* PENGENALAN BUDAYA BATIK INDONESIA BERBASIS *J2ME*

SKRIPSI

Untuk memenuhi sebagian persyaratan untuk mencapai gelar Sarjana Komputer

Disusun oleh :

NOVELIA KHARISMA SETYAYU

NIM. 0910681011

KEMENTERIAN PENDIDIKAN DAN KEBUDAYAAN

UNIVERSITAS BRAWIJAYA

PROGRAM TEKNOLOGI INFORMASI DAN ILMU KOMPUTER

MALANG

2013

LEMBAR PERSETUJUAN

RANCANG BANGUN *MOBILE GAME* PENGENALAN BUDAYA BATIK INDONESIA BERBASIS *J2ME*

SKRIPSI

Diajukan untuk memenuhi persyaratan memperoleh gelar Sarjana Komputer

Disusun Oleh :

NOVELIA KHARISMA SETYAYU

NIM. 0910681011

Skripsi ini telah disetujui oleh dosen pembimbing pada tanggal 2 Juli 2013

Dosen Pembimbing I	Dosen Pembimbing II
Eriq M. Adams J., ST., Mkom.	**Wibisono Sukmo W, ST.,MT.**
NIK. 850410 06 1 1 0027	**NIP. 820404 06 1 1 0091**

repository.ub.ac.id

LEMBAR PENGESAHAN

RANCANG BANGUN *MOBILE GAME* PENGENALAN BUDAYA BATIK INDONESIA BERBASIS *J2ME*

SKRIPSI

Diajukan untuk memenuhi persyaratan memperoleh gelar Sarjana Komputer

Disusun Oleh :

Novelia Kharisma Setyayu

NIM. 0910681011

Skripsi ini telah diuji dan dinyatakan lulus pada tanggal 2 Juli 2013

Penguji I	Penguji II
Denny Sagita Rusdianto, Skom, M.Kom	**Aryo Pinandito, ST., M.MT**
NIK. 851124 06 1 1 0250	**-**

Penguji III

Issa Arwani, S., M.Sc.
NIK. 83092206110074

Mengetahui
Ketua Program Studi Teknik Informatika

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

UNIVERSITAS BRAWIJAYA

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, 9 Juli 2013
Mahasiswa

Novelia Kharisma Setyayu
0910681011

KATA PENGANTAR

Alhamdulillahi rabbil 'alamin. Segala puji syukur penulis panjatkan kehadirat Allah SWT atas segala rahmat dan hidayahNya-lah penulis dapat menyelesaikan Skripsi yang berjudul "Rancang Bangun *Mobile Game* Pengenalan Budaya Batik Indonesia Berbasis *J2ME*". Shalawat dan salam atas junjungan besar kita Nabi Muhammad S.A.W. beserta keluarga dan para sahabat sekalian. Skripsi ini disusun untuk memenuhi sebagian persyaratan memperoleh gelar Sarjana Komputer di Program Studi Teknik Informatika Program Teknologi Informasi dan Ilmu Komputer Universitas Brawijaya Malang.

Melalui kesempatan ini, penulis ingin menyampaikan rasa hormat dan terima kasih penulis yang sebesar-besarnya kepada semua pihak yang telah memberikan bantuan-bantuan baik lahir maupun batin selama penulisan tugas akhir ini. Oleh karena itu, pada kesempatan ini penulis ingin menyampaikan rasa hormat dan terima kasih penulis kepada :

1. Ibunda Tri Setyaningsih, Ayahanda Sajudijono, Adik Whindha Kharisma Dhewanty, serta seluruh keluarga yang tiada henti-hentinya memberikan do'a, semangat dan kasih sayang demi terselesaikannya tugas akhir ini.
2. Bapak Eriq M. Adams J. ST, M.Kom dan Bapak Wibisono Sukmo W, ST.,MT. selaku dosen pembimbing selama pelaksanaan skripsi.
3. Bapak Ir. Sutrisno, M.T., Ir. Heru Nurwasito, M.Kom., Himawat Aryadita, S.T., M.Sc., dan Edy Santoso, S.Kom., selaku Ketua, Wakil Ketua 1, Wakil Ketua 2 dan Wakil Ketua 3 Program Teknologi Informasi dan Ilmu Komputer Universitas Brawijaya.
4. Bapak Drs. Marji, M.T. dan Issa Arwani, S.Kom., M.Sc., selaku Ketua dan Sekretaris Program Studi Teknik Informatika Universitas Brawijaya.
5. Seluruh Dosen Teknik Informatika Universitas Brawijaya atas kesediaan membagi ilmu yang bermanfaat kepada penulis.
6. Seluruh Civitas Akademika Teknik Informatika Universitas Brawijaya yang telah banyak memberi bantuan dan dukungan selama penulis menempuh studi di Teknik Informatika Universitas Brawijaya dan selama penyelesaian tugas akhir ini.

7. Kukuh Heru I, S.Kom, Hanas Subakti dan semua teman-teman yang selalu berbagi ilmu yang bermanfaat kepada penulis.
8. Arianty Anggraini, Winda A Irianto, Fauziah Mayasari, Milani Winangga, Rindang M. H, Ari Agustina, Adestiana Rahmawati, Dian Arisandy, Hanifa Vidya Rizanti, Bekti Widyaningsih, Austin Buya Oriza, Fengky Arga P, Ika Kusumaning Putri, Arga Suwasika P.P dan teman-teman yang selalu membantu dan bertukar semangat dengan penulis selama menyelesaikan tugas akhir ini.
9. Sahabat penulis Anggit Listyoningrum, Firda Istiqomah, Feti Diany, Dyah Ayu Novitasari yang selalu memberikan semangat demi terselesaikannya skripisi ini.
10. Teman–teman Konsentrasi Game serta seluruh Angkatan 2009 Teknik Informatika, terima kasih atas segala bantuannya selama menempuh studi di Teknik Informatika Universitas Brawijaya.

Penulis menyadari bahwa dalam penyusunan tugas akhir ini masih banyak kekurangan baik format penulisan maupun isinya. Oleh karena itu, saran dan kritik membangun dari para pembaca senantiasa kami harapkan guna perbaikan bagi tugas akhir selanjutnya. Semoga tugas akhir ini dapat memberikan manfaat bagi semua pihak, Amin.

Malang, Juli 2013

Penulis

repository.ub.ac.id

ABSTRAK

Novelia Kharisma Setyayu. 2013. Rancang Bangun *Mobile Game* Pengenalan Budaya Batik Indonesia Berbasis *J2ME*. Skripsi Program Studi Teknik Informatika, Program Teknologi Informasi dan Ilmu Komputer, Universitas Brawijaya, Malang. Pembimbing : Eriq Muhammad Adams J, S.T, M.Kom dan Wibisono Sukmo W, ST., MT.

Batik adalah salah satu budaya asli Indonesia, secara konvensional batik merupakan gambar ragam hias di atas sehelai kain dengan menggunakan lilin batik (malam), kemudian diteruskan dengan pemberian warna. Batik merupakan salah satu budaya yang harus dilestarikan. Industri perangkat *mobile* dari tahun-ketahun selalu mengalami peningkatan. Salah satu *asset* yang mendukung perkembangan perangkat *mobile* adalah *mobile gaming*. Memanfaatkan teknologi yang ada dan untuk menjaga budaya Indonesia diluar cara konvensional dirancanglah sebuah game pengenalan budaya batik Indonesia yang merupakan *game* dua dimensi (2D) dengan *gameplay* gabungan *genre platformer* dan *puzzle*, yang bertema edukasi pengenalan budaya batik Indonesia. Tujuan dari permainan adalah mengumpulkan potongan *puzzle*, menyusunnya kemudian mendapatkan informasi tentang budaya batik Indonesia. Perancangan meliputi dua tahapan yaitu *game concept design* dan *technical design* kemudian diimplementasi menggunakan *platform J2ME* dengan bahasa pemrograman *Java*. Pengujian aplikasi menggunakan metode *white-box testing* dengan strategi pengujian unit dan pengujian integrasi, serta metode *black-box testing* dengan strategi pengujian validasi. Pengujian performa permainan dilakukan melalui pengujian FPS (*Frame Per Second)*. Dari hasil pengujian didapatkan kesimpulan bahwa game Pengenalan Budaya Batik telah memenuhi kebutuhan fungsional dan non fungsional.

Kata Kunci: *Mobile Game*, J2ME, Batik.

UNIVERSITAS BRAWIJAYA

repository.ub.ac.id

ABSTRACT

Novelia Kharisma Setyayu. 2013. ***Developing Introduction of Indonesian Batik Culture Mobile Game on J2ME Platform. Undergraduate Thesis of Informatic Engineering Study Program, Information Technology and Computer Science Program, Brawijaya University, Malang. Advisor :*** **Eriq Muhammad Adams J, S.T, M.Kom dan Wibisono Sukmo W, ST., MT.**

Batik is one of the indigenous cultures of Indonesia, batik is conventionally decorative image on a piece of cloth using wax batik (night), then continued with color application. Batik is one culture that should be preserved. Mobile device industry year after year always increase. One of the assets that support the development of a mobile device is a mobile gaming. Utilizing existing technology and to keep the culture of Indonesia beyond the conventional way is designed a Introduction of Indonesian Batik Culture game which is two dimensional (2D) game which gameplay is a combination of platformer and puzzle genre which has the education theme about the recognition of Indonesian batik culture. The aim of the game is to gather the pieces of the puzzle, arrange it and then find the information about Indonesian batik culture. The design consists of 2 stages namely, game concept design and technical design. Afterward it is implemented by using J2ME platform with java programming language. To test the application, the researcher uses white-box testing method with the unit testing strategy and integration strategy, and black-box testing method with validity testing strategy. The game performance testing is done by using FPS (frame per second) testing. From the result of the study, it can be concluded that the introduction to batik culture game has fulfilled the functional and non-functional requirements.

Keywords: Mobile Game, J2ME, Batik

UNIVERSITAS BRAWIJAYA

DAFTAR ISI

KATA PENGANTAR ..... **i**
ABSTRAK ..... **iii**
ABSTRACT ..... **iv**
DAFTAR ISI ..... **v**
DAFTAR GAMBAR ..... **viii**
DAFTAR TABEL ..... **x**
DAFTAR LAMPIRAN ..... **xii**
BAB I **1**
PENDAHULUAN **1**
1.1 Latar Belakang 1
1.2 Rumusan Masalah 2
1.3 Batasan Masalah 2
1.4 Tujuan 3
1.5 Manfaat 3
1.6 Sistematika Penulisan 3
BAB II **5**
DASAR TEORI **5**
2.1 *Game* 5
2.2 Batik 5
2.2.1 Batik Pekalongan 6
2.2.2 Batik Tasikmalaya 6
2.2.3 Batik Garut 7
2.2.4 Batik Solo 8
2.2.5 Batik Yogjakarta 9
2.2.6 Batik Cirebon 9
2.3 *Game Concept Design* 10
2.4 *Unified Modelling Language* 10
2.4.1 Diagram *Use Case* 11
2.4.2 Diagram *Class* 13

2.4.3 Diagram *Activity* 16
2.5 Bahasa Pemrograman J2ME 17
2.5.1 *Connected Limeted Device Configuration (CLDR)* 17
2.5.2 *Mobile Information Device Profile (MIDP)* 17
2.5.3 *Midlet* 17
2.6 Pengujian Perangkat Lunak 18
2.6.1 *White-Box Testing* 19
2.6.2 *Black-Box Testing* 19
2.6.3 Pengujian Performa 20
BAB III 21
METODOLOGI DAN PERANCANGAN 21
3.1 Studi Literatur 21
3.2 Perancangan *Game* 22
3.2.1 *Game Design* 22
3.2.2 *Technical Design* 39
3.3 Implementasi *Game* 50
3.4 Pengujian *Game* 50
BAB IV 51
IMPLEMENTASI 51
4.1 Spesifikasi Sistem 51
4.1.1 Spesifikasi Lingkungan Perangkat Keras 51
4.1.2 Spesifikasi Lingkungan Perangkat Lunak 51
4.2 Batasan-Batasan Dalam Implementasi 52
4.3 Implementasi *Art* 52
4.3.1 Implementasi *Art* Karakter 52
4.3.2 Implementasi *Art* Musuh 53
4.3.3 Implementasi *Art* Item Koleksi 53
4.3.4 Implementasi *Art* Dunia Dalam Permainan 54
4.3.5 Implementasi *Art Puzzle* 58
4.3.6 Implementasi *Art* Deskripsi Batik 59
4.4 Implementasi Prosedur Program 61
4.4.1 Prosedur Pegerakan Pemain 62

4.4.2 Prosedur Interaksi Musuh 66
4.4.3 Prosedur Interaksi Item 67
4.4.4 Prosedur Pengecekan Permainan *Puzzle* 69
4.5 Implementasi Antarmuka dan HUD 70
BAB V 76
PENGUJIAN DAN ANALISIS 76
5.1 Pengujian 76
5.1.1 Pengujian Unit 76
5.1.2 Pengujian Integrasi 86
5.1.3 Pengujian Validasi 95
5.1.4 Pengujian Performa 105
5.1.5 Pengujian Terhadap Pengguna 106
5.2 Analisis 108
5.2.1 Analisis Hasil Pengujian Unit 108
5.2.2 Analisis Hasil Pengujian Integrasi 108
5.2.3 Analisis Hasil Pengujian Validasi 109
5.2.4 Analisis Hasil Pengujian Performa 109
5.2.5 Analisis Hasil Pengujian Terhadap Pengguna 109
BAB VI 112
PENUTUP 112
6.1 Kesimpulan 112
6.2 Saran 112
PUSTAKA DP-1

DAFTAR GAMBAR

Gambar 2. 1 Batik Pekalongan 6
Gambar 2. 2 Batik Tasikmalaya 7
Gambar 2. 3 Batik Garut 8
Gambar 2. 4 Batik Solo 8
Gambar 2. 5 Batik Yogyakarta 9
Gambar 2. 6 Batik Cirebon 10
Gambar 2. 7 Contoh Diagram *Use Case* 12
Gambar 2. 8 Contoh Sebuah *Class* 14
Gambar 2. 9 Contoh Sebuah *Interface* 14
Gambar 2. 10 Contoh Diagram *Class* 15
Gambar 2. 11 Contoh Diagram *Activity* 16
Gambar 2. 12 MIDP *Lifecycle* 18
Gambar 3. 1 Diagram Alir Metode Penelitian 21
Gambar 3. 2 Sketsa cerita halaman pertama 25
Gambar 3. 3 Sketsa cerita halaman kedua 25
Gambar 3. 4 Kontrol permainan utama 26
Gambar 3. 5 Kontrol Permainan *Puzzle* 26
Gambar 3. 6 Logo Universitas Brawijaya 28
Gambar 3. 7 Logo PTIIK 29
Gambar 3. 8 Logo Game Lab 29
Gambar 3. 9 *Game flow* 30
Gambar 3. 10 Matrik 37
Gambar 3. 11 Koleksi 38
Gambar 3. 12 Diagram *use case game* Pengenalan Budaya Batik Indonesia 41
Gambar 3. 13 Diagram *Activity* Melihat *Splash Screen* 44
Gambar 3. 14 Diagram *Activity* Melihat Cerita 45
Gambar 3. 15 Diagram *Activity* Melihat Galeri 46
Gambar 3. 16 Diagram *Activity* Memilih *Level* 46
Gambar 3. 17 Diagram *Activity* Menjalankan Permainan Utama 47
Gambar 3. 18 Diagram *Activity* Menjalankan Permainan *Puzzle* 48
Gambar 3. 19 Diagram *Activity* Melihat Credits 49
Gambar 3. 20 Diagram *Activity* Melihat Bantuan 49
Gambar 4. 1 Art Sprite Karakter 53
Gambar 4. 2 *Art Sprite* Musuh 53
Gambar 4. 3 *Art Sprite* Item Koleksi 54
Gambar 4. 4 *Art* Tile Level Satu 54
Gambar 4. 5 *Art* Tile Level Dua 55
Gambar 4. 6 *Art* Tile Level Tiga 55

Gambar 4. 7 *Art* Tile Level Empat 56
Gambar 4. 8 *Art* Tile Level Lima 56
Gambar 4. 9 *Art* Tile Level Enam 57
Gambar 4. 10 *Art* Tile Semua Level 57
Gambar 4. 11 Implementasi Antarmuka Layar Awal 71
Gambar 4. 12 Implementasi Antarmuka Cerita 71
Gambar 4. 13 Implementasi Antarmuka Menu Utama 72
Gambar 4. 14 Implementasi Antarmuka Pilih Level 72
Gambar 4. 15 Implementasi Antarmuka *Gameplay* 72
Gambar 4. 16 Implementasi Antarmuka Pause 73
Gambar 4. 17 Implementasi Antarmuka Permainan *Puzzle* 73
Gambar 4. 18 Implementasi Antarmuka Galeri 73
Gambar 4. 19 Implementasi Antarmuka Deskripsi Batik 74
Gambar 4. 20 Implementasi Antarmuka Cara Bermain 74
Gambar 4. 21 Implementasi Antarmuka Informasi Permainan 74
Gambar 4. 22 Implementasi HUD Permainan Utama 75
Gambar 4. 23 Implementasi HUD Permainan *Puzzle* 75
Gambar 5. 1 Grafik FPS AVERAGE *Game* 106
Gambar 5. 2 Grafik hasil pengujian kuesioner 107

DAFTAR TABEL

Tabel 2. 1 Keterangan Simbol – Simbol Diagram *Use Case* 12
Tabel 2. 2 Keterangan Symbol – Symbol Diagram *Class* 15
Tabel 2. 3 Keterangan Symbol – Symbol Diagram *Activity* 16
Tabel 3. 1 Konsep Utama *Game* 23
Tabel 3. 2 Kebutuhan Teknologi 23
Tabel 3. 3 Dunia Dalam Permainan 27
Tabel 3. 4 Sudut kamera 28
Tabel 3. 5 Rancangan Antarmuka Menu Utama 31
Tabel 3. 6 Rancangan Antarmuka Pilih Level 31
Tabel 3. 7 Rancangan Antarmuka Gameplay 32
Tabel 3. 8 Rancangan Antarmuka *Game Puzzle* 33
Tabel 3. 9 Rancangan Antarmuka Galeri 33
Tabel 3. 10 Rancangan Antarmuka Deskripsi Galeri 34
Tabel 3. 11 Sketsa Karakter Pemain 35
Tabel 3. 12 Sketsa Karakter Musuh Pertama 36
Tabel 3. 13 Sketsa Karakter Musuh Kedua 36
Tabel 3. 14 Sketsa Karakter Musuh Ketiga 37
Tabel 3. 15 Rancangan Level pada *Game* 38
Tabel 3. 16 Identifikasi aktor 39
Tabel 3. 17 Spesifikasi Kebutuhan Fungsional Permainan 40
Tabel 3. 18 Deskripsi diagram *class* 42
Tabel 3. 19 Deskripsi *Interface* 43
Tabel 4. 1 Spesifikasi Lingkungan Perangkat Keras Komputer 51
Tabel 4. 2 Spesifikasi Lingkungan Perangkat Lunak Komputer 52
Tabel 4. 3 Gambar *Art Puzzle* 58
Tabel 4. 4 *Art* Deskripsi Batik 60
Tabel 4. 5 Implementasi Prosedur Pergerakan Pemain 62
Tabel 4. 6 Implementasi Prosedur Interaksi Musuh 66
Tabel 4. 7 Implementasi Prosedur Interaksi Item 68
Tabel 4. 8 Implementasi Prosedur Pengecekan *Puzzle* 69
Tabel 5. 1 Pemodelan *Flow Graph* operasi moveRight() 77
Tabel 5. 2 *Test Case* Pengujian operasi moveRight() 78
Tabel 5. 3 Pemodelan *Flow Graph* operasi moveLeft() 78
Tabel 5. 4 *Test Case* Pengujian operasi moveLeft() 79
Tabel 5. 5 Pemodelan *Flow Graph* operasi jump() 80
Tabel 5. 6 *Test Case* Pengujian operasi jump() 81
Tabel 5. 7 Pemodelan *Flow Graph* operasi hit() 82
Tabel 5. 8 *Test Case* Pengujian operasi hit() 83

Tabel 5. 9 Pemodelan *Flow Graph* operasi kena() 83
Tabel 5. 10 *Test Case* Pengujian operasi kena() 84
Tabel 5. 11 Pemodelan *Flow Graph* operasi tileMatch()..... 85
Tabel 5. 12 *Test Case* Pengujian operasi tileMatch() 86
Tabel 5. 13 Pemodelan *Flow Graph* Operasi Pergerakan Pemain..... 87
Tabel 5. 14 *Test Case* Pengujian Operasi Pergerakan Pemain..... 89
Tabel 5. 15 Pemodelan *Flow Graph* Operasi Interaksi Musuh..... 91
Tabel 5. 16 *Test Case* Pengujian Interaksi Musuh..... 92
Tabel 5. 17 Pemodelan *Flow Graph* Operasi Interaksi Item 92
Tabel 5. 18 *Test Case* Pengujian Operasi Interaksi Item 93
Tabel 5. 19 Pemodelan *Flow Graph* Operasi Pengecekan *Puzzle* 94
Tabel 5. 20 *Test Case* Pengujian operasi Pengecekan *Puzzle*..... 95
Tabel 5. 21 Kasus Uji Validasi 96
Tabel 5. 22 Hasil Uji Validasi..... 101
Tabel 5. 23 Hasil pengujian kinerja dengan melihat FPS 105
Tabel 5. 24 Pengaruh FPS Terhadap kinerja *Game* 106
Tabel 5. 25 Komponen Kuesioner 107
Tabel 5. 26 Umpan balik saran pengguna..... 107

DAFTAR LAMPIRAN

Lampiran 1. Diagram *Class*……L-1

BAB I
PENDAHULUAN

1.1 Latar Belakang

Perangkat *mobile* merupakan alat komunikasi yang popular di kalangan masyarakat. Dalam beberapa tahun terakhir, pasar dan industri perangkat *mobile* berkembang dengan pesat. Di Indonesia, pengguna perangkat *mobile* meningkat lebih dari 10% dari tahun 2010 ke 2011, dengan total pengguna 78% dari penduduk Indonesia di tahun 2011. Industri yang mendorong perkembangan pasar perangkat *mobile* adalah dunia *gaming*, karena pengguna cenderung menggunakan perangkat untuk bermain *game*. Nilai industri perangkat *mobile* dunia diperkirakan mencapai 8 miliar USD tahun 2012 [DWI-12:5]. *Game* adalah adalah aktifitas menyenangkan yang dilakukan dengan aturan tertentu. Salah satu *genre game* yang telah berkembang adalah *platformer*. *Game platformer* adalah *game* yang dalam permainannya menampilkan maskot utama permaianan yang akan meyusuri *platform* atau lingkungan. Maskot pemain mungkin bisa berayun, melompat, memantul dan menyerang [ROG-10:9]. Salah satu sistem yang bisa digunakan untuk membuat aplikasi *mobile game* adalah *Java 2 Micro Edition* (J2ME). J2ME adalah keluarga *java* yang khusus digunakan untuk membuat aplikasi bagi perangkat *mobile* [PUR-10:5]. Salah satu platform J2ME yang saat ini popular dan sedang dikembangkan oleh Nokia adalah Series 40. Series 40 adalah sistem operasi yang paling banyak digunakan di dunia *mobile*, dengan berbagai pilihan model ponsel yang digunakan setiap hari oleh ratusan juta orang di seluruh dunia [ANO-13]. Menurut statistic *Global Stat Counter*, perangkat *mobile* dengan sistem operasi Series 40 seperti Nokia Asha Smartphone saat ini menempati urutan teratas top sistem operasi mobile di Indonesia [AN1-13].

Indonesia memiliki ratusan bahkan ribuan kebudayaan yang menjadi identitas bangsa, mulai dari kesenian, pakaian adat, rumah adat, kain dan masih banyak lagi. Ironisnya, banyak dari kebudayaan Indonesia yang di klaim oleh Negara lain, karena kurangnya pengetahuan tentang kebudayaan itu sendiri. Batik adalah salah satu budaya asli Indonesia, merupakan gambar ragam hias di atas

sehelai kain dengan menggunakan lilin batik (malam), kemudian diteruskan dengan pemberian warna [KAR-10:9]. Batik memiliki corak dan warna menawan yang sarat akan filosofi. Indonesia memiliki beragam jenis batik dari berbagai daerah. Diantara ragam batik yang terkenal adalah Batik Jogja, Batik Solo, Batik Garut, Batik Tasikmalaya, batik Cirebon, Batik Pekalongan, Batik Pamekasan dan masih banyak lagi. Banyak masyarakat Indonesia yang belum terlalu mengenal budaya batik, ragam corak, asal dan makna filosofinya. Batik membutuhkan sarana untuk melestarikannya, agar tetap menjadi kekayaan budaya Indonesia, agar tidak di klaim oleh Negara lain.

Di luar cara konvensional ada banyak cara untuk memperkenalkan budaya batik Indonesia, salah satunya adalah memanfaatkan perkembangan teknologi yang ada. Memanfaatkan pesatnya perkembangan *platform* Series – 40 yang sedang dikembangkan Nokia, serta pesatnya perkembangan *mobile gaming* saat ini, penulis menggagas tentang perancangan dan pembangunan aplikasi *mobile game* sebagai media pengenalan budaya Batik Indonesia berbasis J2ME. *Game* yang dibangun merupakan gabungan dari *game platformer* dan *puzzle*. Dikemas secara menarik menjadi media pengenalan budaya batik.

1.2 Rumusan Masalah

Berdasarkan latar belakang diatas, penulis merumuskan beberapa masalah sebagai berikut :

1. Merancang *game* pengenalan budaya batik Indonesia?
2. Mengimplementasikan *game* pengenalan budaya batik Indonesia dengan *platform* J2ME?
3. Menguji *game* pengenalan budaya batik Indonesia?

1.3 Batasan Masalah

Untuk menghindari adanya kemungkinan semakin berkembangnya masalah, maka penulisan laporan hanya menitikberatkan permasalahan pada beberapa hal, diantaranya adalah :

1. *Game* yang dibangun merupakan *game* dua dimensi (2D).
2. Perancangan dan implementasi *game* hanya dikhususkan pada OS Nokia Asha Smartphone.
3. Game dapat diimplementasikan pada perangkat mobile dengan spesifikasi *full touch* dan ukuran layar 400 x 240.

1.4 Tujuan

Tujuan penulisan sdesikripsi ini adalah untuk dapat merancang dan mengimplementasikan *mobile game* Pengenalan Budaya Batik Indonesia.

1.5 Manfaat

Manfaat dari pembuatan skripsi ini adalah:

1. Bagi penulis:
   a. Mengaplikasikan ilmu yang didapat selama mengikuti perkuliahan di Informatika Program Teknologi Informasi dan Ilmu Komputer Universitas Brawijaya.
   b. Mendapatkan pengetahuan tentang pembuatan *mobile game* berbasis J2ME.

2. Bagi pengguna
   a. Sebagai media pengenalan budaya batik Indonesia.
   b. Membantu para pengembang *game* dalam mengembangkan *game* ponsel berbasis J2ME.

1.6 Sistematika Penulisan

Sistematika penulisan dalam skripsi ini sebagai berikut:

BAB I Pendahuluan

Memuat latar belakang, rumusan masalah, batasan masalah, tujuan, manfaat dan sistematika penulisan.

BAB II Dasar teori

Menguraikan tentang dasar teori dan referensi yang mendasari rancang bangun *mobile game*.

BAB III Metodologi dan Perancangan

Membahas metode yang digunakan dalam penulisan dan *design* perancangan *game*.

BAB IV Implementasi

Membahas tentang implementasi perangkat lunak.

BAB V Pengujian dan Analisis

Memuat proses pengujian dan hasil analisis terhadap *game*.

BAB VI Penutup

Memuat kesimpulan serta saran untuk pengembangan lebih lanjut.

BAB II
DASAR TEORI

Pada bab ini berisi pembahasan tentang teori dasar yang berhubungan dengan perancangan dan pembangunan *game*.

2.1 *Game*

Game adalah media untuk melakukan aktifitas bermain. *Game* dalam penulisan skripsi ini adalah permainan dalam bentuk aplikasi atau perangkat lunak. *Game* dikategorikan menjadi banyak tipe atau yang disebut dengan *genre game* yaitu *action, action-adventure, action-arcade, platformer, stealth, fighting, shooter, adventure, Role-playing game (RPG), simulation, music/rhythm, party, puzzle, sports, strategy etc* [ROG-10:9].

Game Pengenalan Budaya Batik Indonesia adalah *game* yang memiliki *genre* gabungan dari *platformer* dan *puzzle* yang bertema edukasi. *Platformer game* adalah *game* yang dalam permainannya menampilkan maskot utama g yang akan meyusuri *platform* atau lingkungan. Maskot pemain mungkin bisa berayun, melompat, memantul dan menyerang [ROG-10:9]. *Puzzle game* adalah *game* yang didasarkan pada logika dan pola penyelesaian [ROG-10:11]. *Game* edukasi adalah *game* yang didalamnya memuat konten pendidikan. Dalam perancangan *game* ini *game* dibuat dengan konten edukasi yaitu memperkenalkan buaya batik Indonesia.

2.2 Batik

Batik adalah gambar ragam hias di atas sehelai kain dengan menggunakan lilin batik (malam), kemudian diteruskan dengan pemberian warna. Dilihat dari sejarah perkembangan batik, batik ditemukan pertama kali di Jawa, ditulis di atas sebuah gulungan daun lontar yang ditemukan di Jawa sebelah barat daya pada tahun 1520 [KAR-10:9].

2.2.1 Batik Pekalongan

Batik Pekalongan sudah ada sejak tahun 1800 dan berkembang pesat pada rentang waktu 1825-1830 setelah perang Diponegoro berakhir. Batik Pekalongan terkenal sangat berani memadukan berbagai warna. Motif batik Pekalongan yang sangat khas adalah jenis batik daerah pesisir, yaitu ditemukannya gambar-gambar binatang dan tumbuhan laut. Selain itu, motifnya sangat dipengaruhi oleh budaya Demak yang kental akan nuansa Islam, dipadu dengan budaya yang dibawa oleh para pendatang ke daerah ini. Batik pesisir (utara jawa) ini berkesan cerah dan berwarna-warni dengan motif - motif yang sangat beragam. itu merupakan percampuran seni dan budaya berbagai bangsa. Contohnya batik encim dipengaruhi oleh china, batik jlamprang dipengaruhi arab dan india, batik buketan dipengaruhi belanda, dan batik hokokai dipengaruhi jepang [WAH-12:24]. Pusat perdagangan Batik Pekalongan terletak di Pasar Grosir Setono, pasar ini terletak di jalan utama kalur Pantura, yang berbatasan dengan kota Batang. Contoh batik Pekalongan ditunjukkan pada Gambar 2.1.

Gambar 2. 1 Batik Pekalongan

Sumber : [WAH-12:25]

2.2.2 Batik Tasikmalaya

Budaya membatik di wilayah Tasikmalaya sudah ada lebih dari seabad silam. Nuansa Parahyangan tergambar pada motif bunga anggrek dan burung, merak-ngibing, cala-culu, pisang bali, sapujagat, serta Awi Ngarambat. Warna dan Motif batik Tasikmalaya kuat dan cerah, seperti oranye, merah, hijau, biru, kecuali daerah Sukapura yang banyak menggunakan warna tanah, seperti merah dan kecoklatan. Motif batik umumnya adalah flora dan fauna, misalnya bunga

anggrek, buah kopi, burung merak ngibing (menari), burung bangau dan lain-lain [WAH-12:28]. Salah satu sentra kerajinan batik yang terkenal di daerah Tasikmalaya terletak di Desa Sukapura. Contoh batik Tasikmalaya ditunjukkan pada Gambar 2.2.

Gambar 2. 2 Batik Tasikmalaya

Sumber : [WAH-12:29]

2.2.3 Batik Garut

Setelah pemerintahan republik Indonesia berdiri sekitar tahun 1949, usaha pembatikan di Kabupaten Garut tumbuh kembali dan dikerjakan oleh beberapa keluarga pengrajin. Motif "garutan" umumnya terinspirasi dari alam, misalnya bulu ayam, kelereng, atau kalijagat (sungai) warnanya lebih cerah dari warna batik di jawa tengah. uniknya nama batik bisa diminta sesuai dengan nama pemesan [WAH-12:30].

Ciri khas batik Garut adalah ragam hias yang digunakan naturalistis, merupakan gambaran tentang kehidupan alam nyata atau pengungkapan dan ciri-ciri bentuk alam dan kejadian sehari-hari. Warna khas latar kebanyakan gumading (mangga muda) [KAR-10 : 21]. Contoh batik Garut ditunjukkan pada Gambar 2.3.

Gambar 2. 3 Batik Garut

Sumber : [KAR-10:22]

2.2.4 Batik Solo

Batik Solo sudah ada terlebih dahulu sebelum batik Yogyakarta. Sikap politik yang berbeda dari pihak keraton Solo dan keraton Yagyakarta terhadap Balanda mempengaruhi karakter batik dari keduanya. Sikap keraton Solo yang pro-kolonial tercermin dari batik yang dihasilkannya. Hubungan dekat Solo dan Belanda membuat batik Solo terkesan lebih luwes dan bebas. Ragam hias lebih variatif, dan warna lebih beraneka ragam. Beberapa motif pada batik Solo mengandung filosofi-filosofi yang banyak terpengaruh dari Filafat Jawa dan Hindu. Motif batik Solo yang paling terkenal adalah sidomukti. Batik Solo tidak menggunakan latar belakang putih, warna motif lembut, dan banyak menggunakan motif isen - isen. Perada diterapkan hanya pada garis luar motif atau isen - isen sehingga batik berkesan anggun. Ragam motif batik solo sama atau mirip dengan batik jogja [WAH-12:17]. Sentra penghasil batik di kota Solo yang terkenal yaitu Kampung Laweyan. Kampung Laweyan sudah dikenal sebagai kampung batik semenjak abad ke-19. Contoh batik Solo ditunjukkan pada Gambar 2.4.

Gambar 2. 4 Batik Solo

Sumber : [WAH-12:18]

2.2.5 Batik Yogjakarta

Cikal bakal batik Yogyakarta bermula pada abad ke-18, tepatnya tahun 1755 saat kerajaan Mataram terbelah dua berdasar Perjanjian Giyanti. Pangeran Mangkubumi membawa banyak kain batik dari Kasunanan Solo ke Keraton Ngayogyakarta Hadiningrat. Latar belakang Batik Yogyakarta berwarna putih, warna motif terang atau kontras, tetapi jarang menggunakan isen-isen penuh dan jika menggunakan perada diterapkan pada seluruh motif sehingga batik berkesan mewah. ragam motifnya seputar garis silang (motif ceplok dan kawung), garis miring (motif lereng atau parang), dan anyaman atau limaran. ada pula motif nongeometris, misalnya bunga, burung, naga, awan, gunung dan sebagianya [WAH-12:14]. Sentra kerajinan batik yang paling terkenal di Yogyakarta adalah Desa Wukirsari di Bantul. Contoh batik Yogyakarta ditunjukkan pada Gambar 2.5.

Gambar 2. 5 Batik Yogyakarta
Sumber : [WAH-12:16]

2.2.6 Batik Cirebon

Keterampilan membatik masyarakat Cirebon bermula dari Ki Gede Trusmi, anak buah Sunan Gunung Jati. Ada dua jenis batik Cirebon, yaitu batik Keraton atau klasik dan batik Cirebon Pesisiran. Ciri Batik klasik Cirebon yaitu warna dasar kain lebih muda daripada warna garis pada motif utamanya. Batik klasik Cirebon memiliki warna kuning (sogan gosok), hitam, dan warna dasar krem, atau bisa juga berwarna biru tua, merah tua, dengan warna dasar kain putih gading atau krem. Batik Cirebon Pesisiran memiliki warna-warna yang lebih berani, terang, dan mencolok. Latar kain biasanya bersih dari noda warna yang

tidak dikehendaki pada saat proses pewarnaan. Motif Batik Cirebon tergolong mencolok, mirip lukisan atau kain hiasan dinding dengan jajaran awan, taman, kolam, ikan, anjungan, bangunan dan kaligrafi. Semua itu merupakan perpaduan elemen hindu, china, dan islam. Contoh motif batik Cirebon adalah motif megamendung yang menjadi ciri khas dan terinspirasi dari gaya china [WAH-12: 20]. Desa Trusmi di kecamatan Plered, merupakan pusat industri batik Cirebon. Demikian juga desa-desa di sekitar Trusmi, yang meliputi Desa Gamel, Kaliwulu, Wotgali, Panembaha, dan Kalitengah. Contoh batik Cirebon ditunjukkan pada Gambar 2.6.

Gambar 2. 6 Batik Cirebon

Sumber : [WAH-12:22]

2.3 *Game Concept Design*

Tahap *game concept design* dilakukan dengan pembuatan *game design document*. Pada tahap pembuatan *game design document*, langkah yang pertama kali dilakukan adalah mencari ide. Pencarian ide dapat dilakukan dengan menuliskan ide yang ada atau berdiskusi dengan kelompok orang yang memiliki disiplin ilmu yang berbeda. Kemudian dari ide-ide yang didapatkan akan ditulis pada sebuah catatan yang disebut dengan *brainstorming note* [ROG-10:32]. *Brainstroming note* disusun dalam sebuah *game design document* agar lebih terstruktur sehingga pembaca awam mengerti ringkasan tentang *game* yang dibuat [ROG-10:60].

2.4 *Unified Modelling Language*

Unified Modelling Language (UML) adalah salah satu alat bantu di dunia pengembangan sistem berorientasi obyek. UML menyediakan bahasa pemodelan

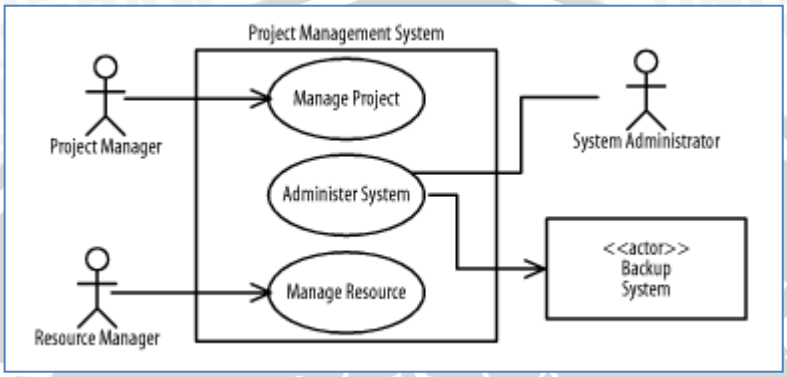
visual yang memungkinkan pengembang sistem membuat cetak biru atas visi mereka dalam bentuk yang baku, mudah dimengerti dan dilengkapi dengan mekanisme yang efektif. UML merupakan kesatuan dari bahasa pemodelan yang dikembangkan oleh Booch, *Object Modeling Technique* (OMT) dan Object *Oriented Software Engineering* (OOSE). Metode Booch dari Grady Booch sangat terkenal dengan nama metode *Design Object Oriented*. Metode ini menjadikan proses analisis dan design ke dalam empat tahapan iteratif, yaitu identifikasi kelas-kelas dan objek-objek, identifikasi semantik dari hubungan objek dan kelas tersebut, perincian interface dan implementasi. Dengan UML, metode Booch, OMT dan OOSE digabungkan dengan membuang elemen-elemen yang tidak praktis ditambah dengan elemen-elemen dari metode lain yang lebih efektif dan elemen-elemen baru yang belum ada pada metode terdahulu sehingga UML lebih ekspresif dan seragam. UML adalah hasil kerja dari konsorsium berbagai organisasi yang berhasil dijadikan sebagai standart baku dalam OOAD (Object *Oriented Analysis and Design).* UML mempunyai sejumlah elemen grafis yang bisa dikombinasikan menjadi diagram. Karena ini merupakan sebuah bahasa, UML mempunyai sejumlah aturan yang mengkombinasikan element – element [MUN-05:22].

Dalam UML pemodelan digambarkan dengan penggunaan diagram-diagram yang masing-masing memiliki fungsi dan makna sendiri-sendiri. Diagram-diagram ini dapat digunakan sesuai dengan kebutuhan saat kita akan merancang perangkat lunak. Beberapa diantaranya adalah diagram *use case*, diagram *class* dan diagram *activity*.

2.4.1 Diagram *Use Case*

Use case adalah deskripsi fungsi dari sebuah sistem dari perspektif pengguna. *Use case* bekerja dengan cara mendeskripsikan tipikal interaksi antara user (pengguna) sebuah sistem dengan sistemnya sendiri melalui sebuah cerita bagaiman sebuah sistem dipakai. Urutan langkah-langkah yang menerangkan antara pengguna dan sistem disebut scenario. Setiap scenario mendeskripsikan urutan kejadian. Setiap urutan diinisialisasi oleh orang, sistem yang lain, perangkat keras atau urutan waktu. Dengan demikian secara singkat bisa

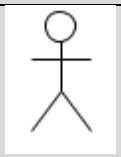
dikatakan use case adalah serangkaian scenario yang digabungkan bersama-sama oleh tujuan umum pengguna [MUN-05:63]. Contoh Use Case diagram ditunjukkan pada gambar 2.7. Keterangan simbol – simbol diagram *use case* ditunjukkan pada Tabel 2.1.

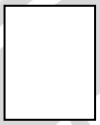
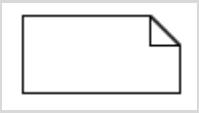


Gambar 2. 7 Contoh Diagram *Use Case*

Sumber : [ALH-03]

Tabel 2. 1 Keterangan Simbol – Simbol Diagram *Use Case*

NO	GAMBAR	NAMA	KETERANGAN
1		*Actor*	Menspesifikasikan himpunan peran yang pengguna mainkan ketika berinteraksi dengan *use case*.
2		*Dependency*	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri *(independent)* akan mempengaruhi elemen yang bergantung padanya sebagai elemen yang tidak mandiri (*independent*).
3		*Generalization*	Hubungan dimana objek anak (*descendent*) berbagi perilaku dan struktur data dari objek yang ada di atasnya (objek induk / *ancestor*).
4		*Include*	Menspesifikasikan bahwa *use case* sumber secara *eksplisit*.

5		*Extend*	Menspesifikasikan bahwa *use case* target memperluas perilaku dari *use case* sumber pada suatu titik yang diberikan.
6		*Association*	Apa yang menghubungkan antara objek satu dengan objek lainnya.
7		*Sistem*	Menspesifikasikan paket yang menampilkan sistem secara terbatas.
8		*Use Case*	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu actor.
9		*Collaboration*	Interaksi aturan-aturan dan elemen lain yang bekerja sama untuk menyediakan perilaku yang lebih besar dari jumlah dan elemen-elemennya (sinergi).
10		*Note*	Elemen fisik yang eksis saat aplikasi dijalankan dan mencerminkan suatu sumber daya komputasi.

Sumber : Dasar Teori

2.4.2 Diagram *Class*

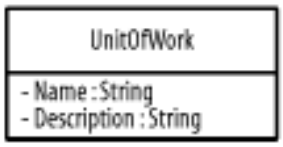
Class adalah sebuah spesifikasi yang jika diinstansiasi akan menghasilkan sebuah objek dan merupakan inti dari pengembangan dan desain berorientasi objek. *Class* dalam notasi UML digambarkan dengan kotak. Nama *class* menggunakan huruf besar di awal kalimat dan diletakkan diatas kotak. Bila class mempunyai nama yang terdiri dari 2 suku kata atau lebih, maka semua suku kata digabungkan tanpa spasi dengan huruf awal tiap suku kata menggunanan huruf besar [MUN-05:35]. Diagram c*lass* menggambarkan struktur dari *class, package* dan *object* yang saling berhubungan satu sama lain seperti *containment,*

pewarisan, asosiasi, dan lain-lain. Untuk contoh sebuah *class* ditunjukkan pada Gambar 2.8.

Class dapat merupakan implementasi dari sebuah *interface*, yaitu *class* abstrak yang hanya memiliki metoda. *Interface* adalah *classifier* yang menyatakan dari serangkaian fitur umum yang koheren dan wajib di implementasikan. Contoh sebuah interface ditunjukkan pada Gambar 2.9. Hubungan antar *class* adalah sebagai berikut :

1. Asosiasi, yaitu hubungan statis antar *class*. Umumnya menggambarkan *class* yang memiliki atribut berupa *class* lain, atau *class* yang harus mengetahui eksistensi *class* lain. Panah *navigability* menunjukkan arah *query* antar *class*.
2. Agregasi, yaitu hubungan yang menyatakan bagian ("terdiri atas..").
3. Pewarisan, yaitu hubungan hirarkis antar *class*. *Class* dapat diturunkan dari *class* lain dan mewarisi semua atribut dan metoda *class* asalnya dan menambahkan fungsionalitas baru, sehingga ia disebut anak dari *class* yang diwarisinya. Kebalikan dari pewarisan adalah generalisasi.

Contoh diagram *class* ditunjukkan pada gambar 2.10. Keterangan simbol – simbol diagram *class* ditunjukkan pada Tabel 2.2.

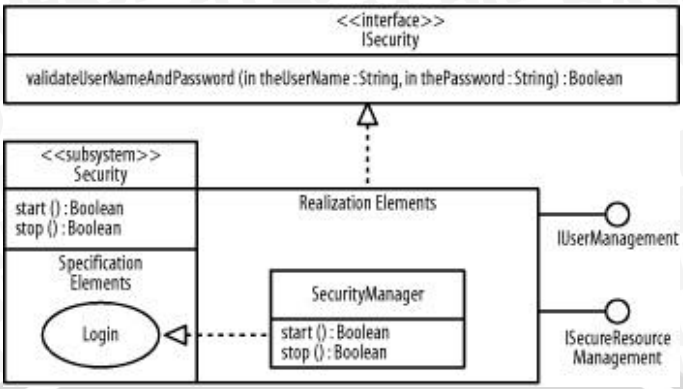


Gambar 2. 8 Contoh Sebuah *Class*

Sumber : [ALH-03]

Gambar 2. 9 Contoh Sebuah *Interface*

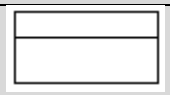
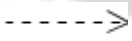
Sumber : [ALH-03]



Gambar 2. 10 Contoh Diagram *Class*

Sumber : [ALH-03]

Tabel 2. 2 Keterangan Symbol – Symbol Diagram *Class*

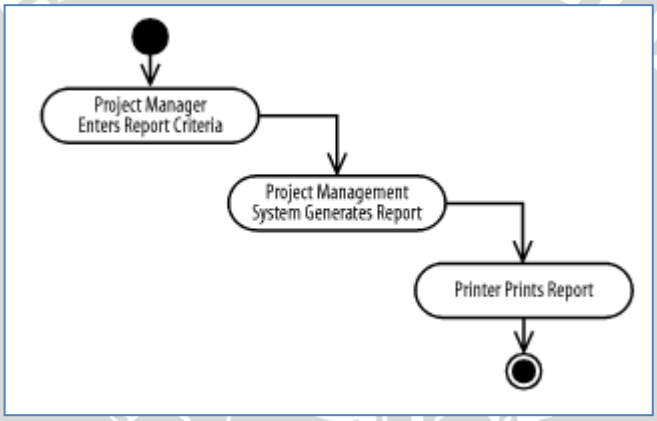
NO	GAMBAR	NAMA	KETERANGAN
1		*Generalization*	Hubungan dimana objek anak *(descendent)* berbagi perilaku dan struktur data dari objek yang ada di atasnya (objek induk / *ancestor*).
2		*N-ary Association*	Upaya untuk menghindari asosiasi dengan lebih dari 2 objek.
3		*Class*	Himpunan dari objek-objek yang berbagi atribut serta operasi yang sama.
4		*Collaboration*	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu aktor.
5		*Realization*	Operasi yang benar-benar dilakukan oleh suatu objek.
6		*Dependency*	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri *(independent)* akan mempengaruhi elemen yang bergantung padanya atau elemen yang tidak mandiri.
7		*Association*	Apa yang menghubungkan antara objek satu dengan objek lainnya.

8	1..*, 0...*, 1...1	*Multiplicity*	Menunjukkan jumlah objek yang diperbolehkan menurut logika.

Sumber : Dasar Teori

2.4.3 Diagram *Activity*

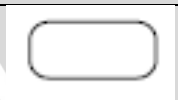
Diagram *Activity* adalah diagram yang menggambarkan kegiatan dan tanggung jawab unsur-unsur yang membentuk sistem. Contoh diagram *activity* di tunjukkan pada Gambar 2.11. Keterangan simbol – simbol diagram *activity* ditunjukkan pada Tabel 2.3.



Gambar 2. 11 Contoh Diagram *Activity*

Sumber : [ALH-03]

Tabel 2. 3 Keterangan Symbol – Symbol Diagram *Activity*

NO	GAMBAR	NAMA	KETERANGAN
1		*Simple State*	Sebuah state tanpa substruktur
4		*Initial State*	Menggambarkan penghubung antara boundary dengan table: Kelas yang mewakili objek yang melakukan proses pengolahan informasi.
5		*Final State*	Menggambarkan tempat mulai dan berakhirnya pengiriman sebuah pesan
6		*Junction State*	Menggambarkan pengiriman pesan.

2.5 Bahasa Pemrograman J2ME

Java 2 Micro Edition atau yang biasa disebut J2ME adalah lingkungan pengembangan yang didesain untuk meletakkan perangkat lunak *Java* pada barang elektronik berserta perangkat pendukungnya. J2ME membawa *Java* ke dunia informasi, komunikasi, dan perangkat komputasi yang lebih kecil dibandingkan dengan computer desktop. J2ME biasa digunakan pada telepon selular, pager, PDA, dan sejenisnya [DWI-12:26].

Teknologi J2ME juga memiliki beberapa keterbatasan jika diaplikasikan pada ponsel. J2ME sangat bergantung pada device yang digunakan, bisa dari merk ponsel, maupun kemampuan ponsel, dan dukungan terhadap teknologi J2ME [DWI-12:28].

2.5.1 *Connected Limeted Device Configuration (CLDR)*

CLDC atau *Connected Limited Device Cofiguration* adalah perangkat dasar dari J2ME, spesifikasi dasar yang berupa *library* atau API yang diimplementasikan pada J2ME, seperti yang digunakan pada telepon selular, pager, dan PDA. Perangkat tersebut dibatasi dengan keterbatasan memori, sumber daya, dan kemampuan memproses. Spesifikasi CLDC pada J2ME adalah spesifikasi minimal pada package, class, dan sebagian fungsi Java Virtual Machine yang dikurangi agar dpat diimplementasikan dengan keterbatasan sumber daya pada alat tersebut, JVM yang digunakan adalah KVM (Kilobyte Virtual Machine) [DWI-12:28].

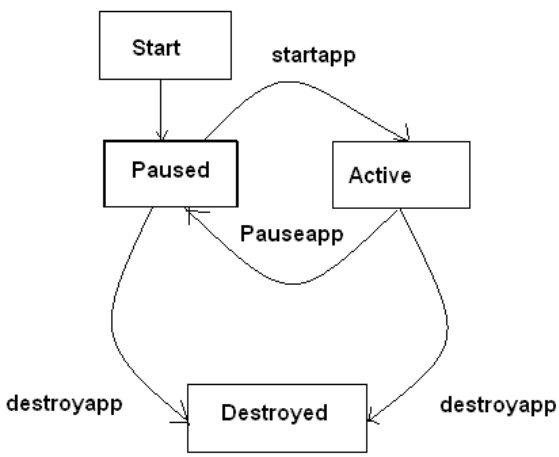
2.5.2 *Mobile Information Device Profile (MIDP)*

MIDP atau *Mobile Information Device Profile* adalah spesifikasi untuk sebuah profil J2ME. MIDP memilik lapisan diatas CLDC, API tambahan untuk daur hidup aplikasi, antar muka, jaringan, dan pemyimpanan persisten [DWI-12:28].

2.5.3 *Midlet*

Suatu aplikasi *Mobile Information Device Profile* (MIDP) pada J2ME disebut MIDlet. MIDlet adalah bagian dari *package* `javax.microedition`.

`midlet`. Sebuah MIDlet harus *di-extend* dengan *class* MIDlet. MIDlet terdiri dari beberapa method yang harus ada, yaitu `contructor()`, `startApp()`, `pauseApp()`, `destroyApp()`. Method yang pertama adalah `startApp()` dipanggil oleh *Application Management Software* (AMS) ketika MIDlet dijalankan. Method yang kedua adalah `pauseApp()`, yang akan di-*invoke* oleh AMS ketika dalam situasi tertentu, seperti pengguna menerima telepon ketika bermain *game*, peringatan *low battery*, dan lain-lain. Dan dapat digunakan untuk meminimasi CPU, memory, atau konsumsi baterenya, ketika *game* kita tidak dalam *active state*. *Method* yang terakhir adalah `destroyApp()`. Method ini dipanggil untuk *destroy* MIDlet oleh AMS. Dalam *method* ini harus *clean-up resource, action,* dan lain-lain. Dan memberitahukan AMS untuk menyempurnakan *cleanup* dengan `notifDestroy()`. Ketiga method di atas merupakan bagian dari MIDP *Lifecycle* yang ditunjukkan pada Gambar 2.12 [DWI-12:29].



Gambar 2. 12 MIDP *Lifecycle*

Sumber : [DWI-12:30]

2.6 Pengujian Perangkat Lunak

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 ini menyediakan pendekatan sistematis untuk pengujian oleh *developer*. Terlebih lagi metode ini menyediakan mekanisme yang dapat membantu memastikan

kelengkapan dari pengujian dan menyediakan kemungkinan tertinggi untuk menemukan kesalahan-kesalahan dalam perangkat lunak. Teknik atau metode perancangan kasus uji yang digunakan adalah *black-box testing* dan *white-box testing* [PRE-10:443].

2.6.1 *White-Box Testing*

White box testing adalah teknik pengujian yang menguji berdasarkan jalur internal, struktur dan implementasi dari perangkat lunak yang sedang diuji. *White box testing* adalah teknik pengujian yang menggunakan struktur kontrol dari prosedur yang terdapat dalam perancangan untuk membuat kasus uji. Ada dua jenis pengujian yang termasuk *white box testing* yaitu *basis path testing* dan *control structure testing*. *Basis path testing* adalah teknik dalam metode *white box testing* untuk menyediakan desain *test case* yang dapat digunakan untuk menghitung kompleksitas desain prosedural untuk menghitung banyaknya jalur eksekusi. *Test case* tersebut digunakan untuk menjamin apakah tiap statement telah dijalankan minimal satu kali tiap pengujian. Dalam konteks *basis path*, nilai yang dihitung di dalam *cyclomatic complexity* menjelaskan banyaknya jumlah jalur independent yang harus disediakan dalam pengujian *basis path* [PRE-10:445].

2.6.2 *Black-Box Testing*

Black box testing adalah teknik pengujian yang menguji hanya berdasarkan kebutuhan dan spesifikasi. *Black box testing* juga disebut sebagai *behavioral testing* dan berfokus pada kebutuhan fungsi dari perangkat lunak [PRE-10:460]. Proses umum yang terjadi pada *black box testing* yaitu:

a. Kebutuhan atau spesifikasi dianalisa terlebih dahulu.
b. Penentuan input valid terpilih berdasarkan spesifikasi untuk menentukan perangkat lunak berjalan dengan benar. Input yang tidak valid juga harus dipilih untuk memverifikasi bahwa perangkat lunak dapat mendeteksinya dan menanganinya dengan baik.

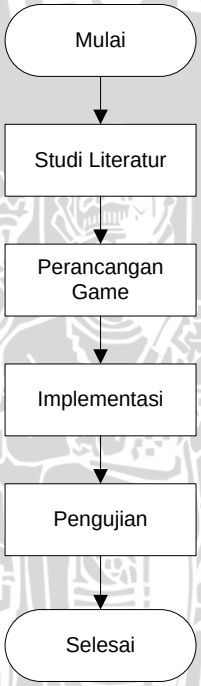
c. Penentuan output yang diharapkan sesuai dengan input yang telah dipilih.
d. Pengujian dibuat dengan input yang telah dipilih.
e. Pengujian dijalankan.
f. Output yang sebenarnya dibandingkan dengan output yang diharapkan
g. Penentuan dibuat menyangkut perangkat lunak berfungsi sesuai dengan spesifikasi yang telah ditentukan.

2.6.3 Pengujian Performa

Pengujian ini dilakukan dengan melihat *frame rate* (*Frame Per Second*), merupakan jumlah *frame* animasi bergerak yang ditampilkan dalam setiap detik. Misal dalam kartun Bugs Bunny, normalnya kartun berkedip dengan 16 FPS per detik. Ini cukup cepat untuk mata manusia percaya bahwa gambar bergerak. Dalam permainan untuk mencapai 16 FPS, perangkat perlu melukis setiap frmae selama 62,5 milidetik. Namun permainan tidak sedetail kartun. Karakter permainan pada J2ME biasanya tidak terlalu berubah banyak per frame sehingga *frame rate* 10 FPS dapat diterima [FOX-02:175].

BAB III
METODOLOGI DAN PERANCANGAN

Pada bab ini dijelaskan mengenai metodologi yang dilakukan dalam penelitian serta perancangan *Game* Pengenalan Budaya Batik Indonesia. Diagram alir metodologi penelitian ditunjukkan pada gambar 3.1. Perancangan yang terdiri atas dua tahap yaitu *Game Design* dan *Technical Design* dijelaskan pada sub bab selanjutnya.



Gambar 3. 1 Diagram Alir Metode Penelitian

3.1 Studi Literatur

Sebelum perancangan *game* akan dilakukan studi literatur terlebih dahulu untuk memahami konsep-konsep yang harus dipelajari.
Literatur tersebut antara lain:

1. *Game*
2. Batik
3. *Game Concept Design*
4. *Unified Modelling Language*

a. Use Case Diagram

b. Class Diagram

c. Sequence Diagram

5. Bahasa Pemrograman J2ME.
6. Pengujian Perangkat Lunak
    a. *White-Box Testing*
    b. *Black-Box Testing*
    c. Pengujian Performa

3.2 Perancangan *Game*

Tahap perancangan *game* terdiri dari dua bagian yaitu *game design* dan *technical design.*

3.2.1 *Game Design*

Tahap *game concept design* dilakukan dengan pembuatan *game design document*. Pada tahap pembuatan *game design document,* langkah yang pertama kali dilakukan adalah mencari ide. Pencarian ide dapat dilakukan dengan menuliskan ide yang ada atau berdiskusi dengan kelompok orang yang memiliki disiplin ilmu yang berbeda. Kemudian dari ide-ide yang didapatkan akan ditulis pada sebuah catatan yang disebut dengan *brainstorming note* [ROG-10:32]. Pada bab ini dijelaskan pembuatan *game design document* yang merupakan detail-detail dalam permainan. Tujuan dibuatnya dokumen ini adalah untuk mengkomunikasikan kepada tim untuk meminimalisir kesalahan pada saat implementasi karena kurang mengerti tentang game yang dibuat [ROG-10:75].

Game design documenten mencantumkan segala rincian dalam desain pembuatan *game*. Rincian tersebut meliputi Konsep utama *game,* Kebutuhan Teknologi, Tujuan *game*, Cerita, Deskripsi *Gameplay*, Kontrol Permainan, Dunia Dalam Permainan, Sudut Kamera, *Splash Screen*, *Gameflow,* Antarmuka, Karakter Pemain, Musuh, Matrik, Koleksi, Nyawa, Level, Musik dan Suara.

1. Konsep Utama *Game*

Konsep utama *game* ditunjukkan pada Tabel 3.1.

Tabel 3. 1 Konsep Utama *Game*

No	Elemen	Keterangan
1.	Judul Permainan	Batik Mania
2.	*Platform*	J2ME
3.	Target Device	Nokia SDK 2.0
4.	Target Usia	Semua usia
5.	*Genre game*	Platformer – *Puzzle* – Education
6.	Produk Sejenis	*Braid*
7.	*USP(Unique Selling Point)*	Media edukasi untuk memperkenalkan budaya batik Indonesia. Gabungan dua *genre game* dalam satu aplikasi. Art 2D yang menarik. Didukung fungsi *touch screen*. Didukung layar yang lebih besar dibandingkan *mobile java* lainnya. Single mode.

2. Kebutuhan Teknologi

Daftar kebutuhan teknologi untuk membangun *game* ditunjukkan pada Tabel 3.2.

Tabel 3. 2 Kebutuhan Teknologi

No	Kebutuhan	Kegunaan
1.	Aplikasi pengolah grafis	Sebagai media gambar digital.
2.	Aplikasi Netbean	Untuk membagun aplikasi J2ME.
3.	SDK 2.0 Nokia	Sebagai perangkat lunak tambahan agar komatibel dengan target pembangunan.
4.	Ponsel Nokia	Sebagai alat uji perangkat lunak.

3. Tujuan *Game*

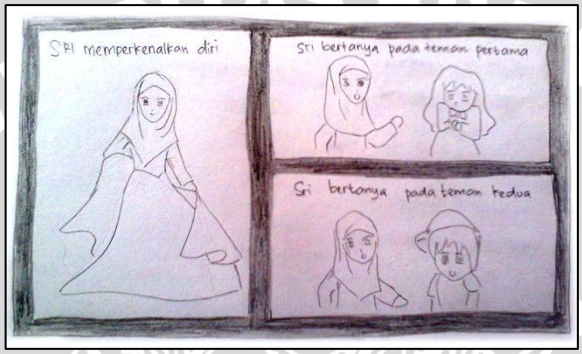
Tujuan utama dari *game* ini adalah menyelesaikan misi permainan untuk mengumpulkan potongan-potongan *puzzle* batik kemudian disusun menjadi gambar yang utuh untuk mendapatkan informasi tentang batik pada suatu daerah.

4. Cerita

Rancangan jalan cerita dalam permainan ini di susun dalam bentuk komik sebanyak dua halaman. Halaman pertama mula-mula memperkenalkan tokoh Sri yang merupakan pemain utama dalam permainan. Dikisahkan Sri adalah seorang gadis Jawa yang sangat mencintai budaya Indonesia, khusunya batik. Dia ingin melestarikan budaya batik dengan mencari informasi tentang batik dari berbagai daerah di Indonesia. Karena itu dia memutuskan untuk bertanya kepada teman-temannya. Dia mulai bertanya pada temanya yang pertama bernama Sinta tentang batik dari beberapa daerah di Indonesia, namun Sinta tidak terlalu mengetahui tentang batik daerah-daerah tersebut. Kemudian Sri bertanya kepada temannya yang kedua bernama Joko. Namun Sri pun mendapat jawaban yang sama. Sketsa cerita halam kedua ditunjukkan pada Gambar 3.2.

Halaman kedua merupakan lanjutan dari halaman pertama. Tanpa lelah Sri terus berusaha. Dia bertanya kepada temannya yang ketiga, bernama Ratna. Namun Sri malang mendapat jawaban yang sama. Sri mulai marah kepada teman-temannya. Bagaimana kita bisa menjaga budaya negeri sendiri jika kita tidak benar-benar paham tentang budaya tersebut. Sri sangat sedih, dia takut jika karena ketidaktahuan warga Indonesia tentang budaya, akan semakin banyak budaya yang di klaim negara lain. Untuk itu Sri bertekat untuk menjalankan misi mencari tau tentang budaya dan petualanganpun dimulai. Sketsa cerita halam kedua ditunjukkan pada Gambar 3.3.

Gambar 3. 2 Sketsa cerita halaman pertama



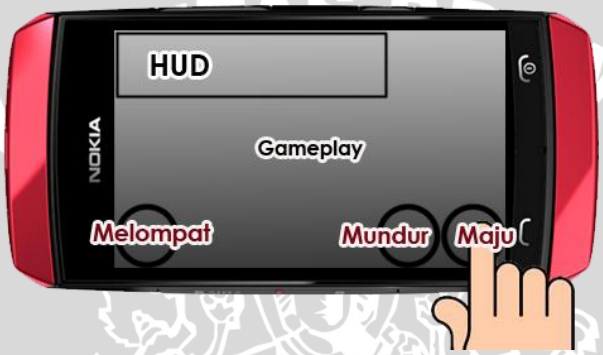
Gambar 3. 3 Sketsa cerita halaman kedua

5. Deskripsi *Gameplay*

Batik Mania adalah *game* dua dimensi dengan penggabungan *genre platformer* dan *puzzle* dengan konten edukasi tentang pengenalan budaya batik. Pemain mengendalikan tokoh Sri (pemain). Sri berlari menyusuri dunia batik untuk mengumpulkan potongan-potongan *puzzle* dan bonus nyawa, tapi di perjalanan Sri akan menghadapi banyak tantangan dari Buto (musuh). Buto datang dari arah berlawanan secara acak sesuai line jalan. Apabila berhadapan dengan pemain, musuh dapat membunuh pemain dengan cara memukul, sedangkan pemain dapat membunuh musuh dengan cara melompat dan menimpa dari atas. Setiap terserang satu kali pemain akan kehilangan sebagian nyawa dan begitu selanjutnya. Setelah berhasil mengumpulkan seluruh potongan *puzzle*, pemain akan masuk ke dalam permainan *puzzle* dan menyusun *puzzle* menjadi gambar yang benar. Apabila pemain telah berhasil melewati game *puzzle* barulah pemain mendapat bonus informasi batik sesuai level yang dipilih.

6. Kontrol *Game*

Kontrol *game*, semuanya di lakukan dengan cara menyentuh layar (*touch screen*). Kontrol pemain pada permainan utama adalah menyentuh tombol yang telah disediakan untuk menggerakkan pemain. Kontrol permainan utama ditunjukkan pada gambar 3.4. Kontrol permainan *puzzle* adalah menyentuh potongan pertama diikuti potongan kedua untuk merubah posisinya. Kontrol permainan *puzzle* ditunjukkan pada gambar 3.5.



Gambar 3. 4 Kontrol permainan utama

Gambar 3. 5 Kontrol Permainan *Puzzle*

7. Dunia Dalam Permainan

Dunia dalam permaian di rancang sedemikan rupa dengan konsep art 2D menyerupai motif -motif batik sehingga pemain serasa berada dalam dunia batik. Rancangan dunia dalam permainan ditunjukkan pada Tabel 3.3.

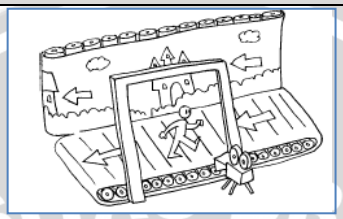
Tabel 3. 3 Dunia Dalam Permainan

No	Nama	Gambar
1.	Awan batik	
2.	Matahari batik	
3.	Tanaman batik	
4.	Bangunan batik	

8. Sudut Kamera

Sudut kamera ditunjukkan pada Tabel 3.4.

Tabel 3. 4 Sudut kamera

	Nama	**Ilustrasi**	**Keterangan**
Kamera *view*	*Side scroll* kamera.		Kamera seolah – olah bergeser kesamping.
Kamera *shot*	*Extreme wide shot* (EWS)		Menampilkan karakter dari sudut pandang yang jauh.
Kamera *angle*	*Eye level*		Objek dari tujuan kamera sejajar dengan pandangan mata.

9. Layar Awal

Layar pembuka permainan yang berisi logo institusi yang menaungi pembuatan *game* meliputi Logo Universitas Brawijaya yang ditunjukkan pada gambar 3.6. Logo PTIIK yang ditunjukkan pada gambar 3.7 dan Logo GameLab yang ditunjukkan pada gambar 3.8.

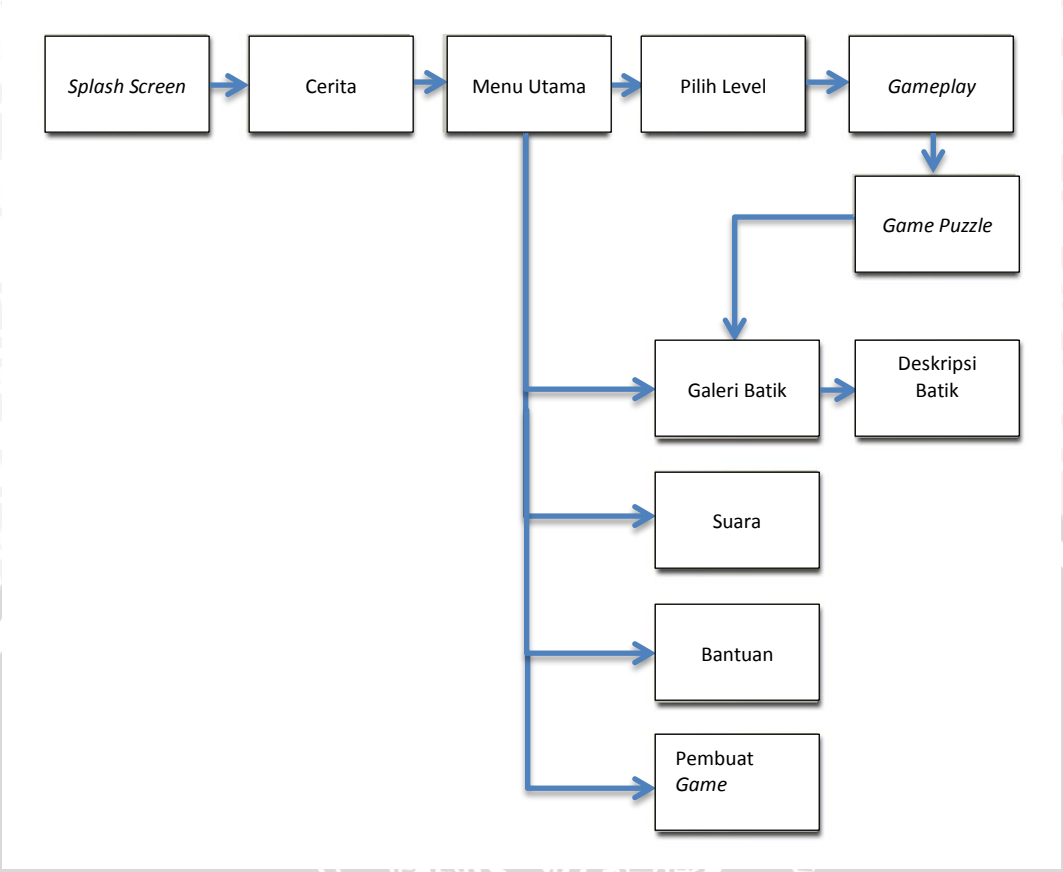
Gambar 3. 6 Logo Universitas Brawijaya

Gambar 3. 7 Logo PTIIK

Gambar 3. 8 Logo Game Lab

10. Alur *Game*

Alur jalan permainan yang dapat dilihat oleh pemain. Terdapat beberapa layar dalam permainan ini meliputi *splash screen*, cerita, menu utama, pilih level, *gameplay, game puzzle*, galeri, deskripsi batik, bantuan dan informasi pembuat *game*. Gambaran *game flow* ditunjukkan pada Gambar 3.9.

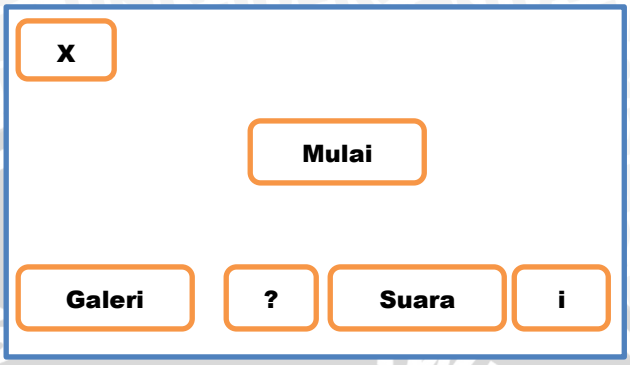


Gambar 3. 9 *Game flow*

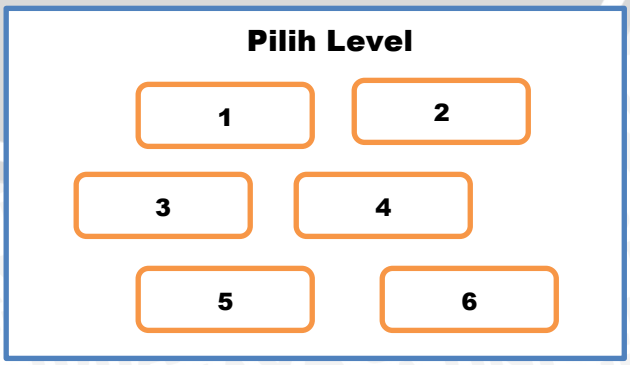
11. Antarmuka

Rancangan antarmuka menu utama ditunjukkan pada Tabel 3.5. Rancangan antarmuka pilih level ditunjukkan pada Tabel 3.6. Rancangan antarmuka *gameplay* ditunjukkan pada Gambar 3.7. Rancangan antarmuka *game puzzle* ditunjukkan pada Gambar 3.8. Rancangan antarmuka galeri ditunjukkan pada Gambar 3.9. Rancangan antarmuka deskripsi galeri ditunjukkan pada Gambar 3.10.

Tabel 3. 5 Rancangan Antarmuka Menu Utama

Nama Layar	**Main Menu**
Sketsa	
Keterangan	Main menu berisi tombol mulai, galeri, bantuan, info, suara dan keluar. 1. Mulai : Mulai permaian. 2. Galeri: Galeri tentang deskripsi batik dari beberapa daerah di Indonesia. 3. Suara : Pengaturan suara *on/off*. 4. i : Informasi nama pembuat permainan, sumber informasi batik dan sumber – sumber terkait. 5. ? : Informasi cara bermain. 6. X : Keluar permainan

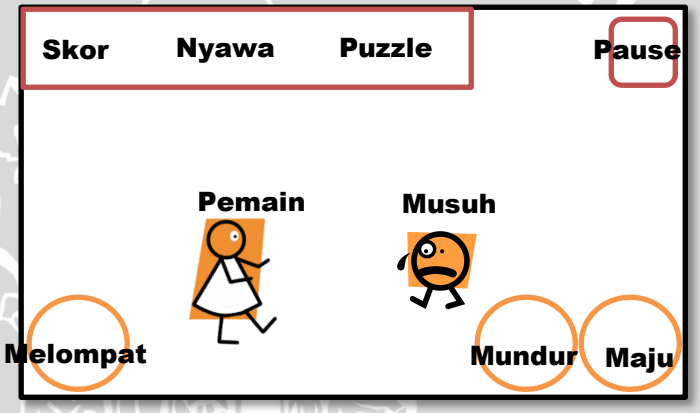
Tabel 3. 6 Rancangan Antarmuka Pilih Level

Nama Layar	**Pilih Level**
Gambar	

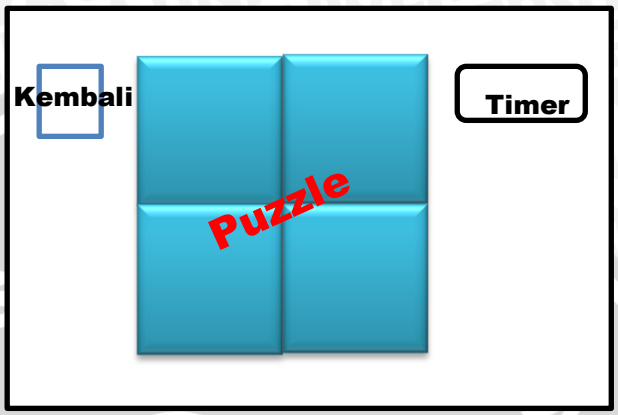
Keterangan	Setelah di tekan tombol mulai maka pemain masuk ke dalam pilih level. Berisi pilihan level mana yang ingin dimainkan. 1. Pekalongan 2. Tasikmalaya 3. Garut 4. Solo 5. Yogyakarta 6. Cirebon

Sumber : Metodologi dan Perancangan

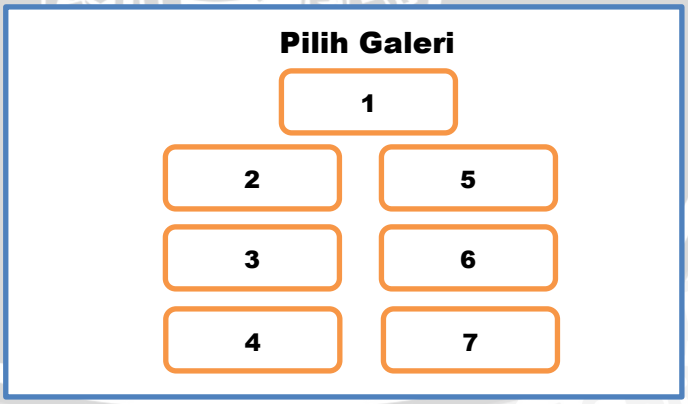
Tabel 3. 7 Rancangan Antarmuka Gameplay

Nama Layar	**Gameplay**
Gambar	
Keterangan	Tampilan jalannya permainan sesuai level yang telah di pilih. 1. Level : menunjukkan level sekarang 2. Skor *:* menunjukkan jumlah skor yang di dapat. 3. Jumlah nyawa : menunjukkan jumlah nyawa yang dimiliki pemain 4. Jumlah icon *puzzle* : menunjukkan jumlah icon *puzzle* yang telah berhasil dikumpulkan pemain 5. Pause : tombol pause yang berisi: menghentikan permainan sesaat, kembali ke main menu atau ulangi permainan 6. Pemain : karakter yang dikendalikan pemain 7. Musuh : karakter yang menyerang pemain

Tabel 3. 8 Rancangan Antarmuka *Game Puzzle*

Nama Layar	***Puzzle***
Gambar	
Keterangan	Tampilan permainan *puzzle* yang ditampilakn ketika pemain berhasil mengumpulkan potongan *puzzle*. 1. *Puzzle* : canvas *puzzle* yang harus di susun. 2. Timer : menunjukkan estimasi waktu 3. Kembali : tombol kembali ke menu awal

Tabel 3. 9 Rancangan Antarmuka Galeri

Nama Layar	**Gallery**
Gambar	
Keterangan	Setelah pemain berhasil menyelesaikan misi permainan, otomatis pemain diberi hak akses untuk melihat isi galleri. Di dalam gallery terdapat tujuh deskripsi yang dapat dipilih pemain. Keterangan : 1. Sejarah

	2. Pekalongan 3. Tasikmalaya 4. Garut 5. Solo 6. Yogyakarta 7. Cirebon 8. Keluar

Tabel 3. 10 Rancangan Antarmuka Deskripsi Galeri

Nama Layar	**Deskripsi Gallery**
Gambar	
Keterangan	Deskripsi gallery berisi gambar contoh batik dari daerah yang ingin diperkenalkan serta deskripsi singkat tentang batik tersebut.

12. Karakter Pemain

Informasi karakter dalam permainan ini ditunjukkan Tabel 3.11.

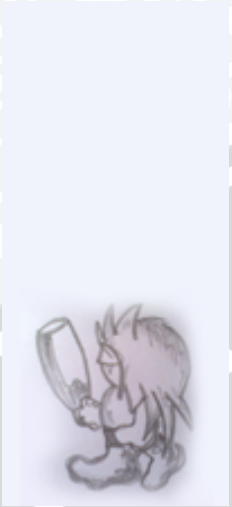
Tabel 3. 11 Sketsa Karakter Pemain

Sketsa	Profil	
	Nama	Sri
	Kelamin	Perempuan
	Usia	17 Tahun
	Kepribadian	Anggun tapi tangguh
	Pakaian	Kebaya batik coklat
	Kebangsaan	Indonesia
	Suku	Jawa
	Keahlian	Berjalan, melompat, menggencet musuh.
	Posisi awal	Menghadap kekanan
	Nyawa	Mula – mula pemain di beri 50 nyawa, sekali hit nyawa pemain akan berkurang 1.

13. Musuh

Informasi musuh pertama ditunjukkan pada Tabel 3.12. Informasi musuh kedua ditunjukkan pada Tabel 3.13. Informasi musuh kedua ditunjukkan pada Tabel 3.14.

Tabel 3. 12 Sketsa Karakter Musuh Pertama

Sketsa	Profil	
	**Nama**	Buto Abang
	Tampilan	Menyeramkan, berwarna hijau, berambut merah panjang, membawa tongkat.
	Ukuran	Kecil
	Kecepatan	Sedang
	Senjata	Tongkat
	Keahlian	Berjalan, memukul
	Cara membunuh	Pemain menimpa dari atas musuh
	Nyawa	1
	Skor	10

Tabel 3. 13 Sketsa Karakter Musuh Kedua

Sketsa	Profil	
	Nama	Buto Kuning
	Tampilan	Menyeramkan, berwarna hijau, berambut kuning panjang, membawa tongkat.
	Ukuran	Medium
	Kecepatan	Sedang
	Senjata	Tongkat
	Keahlian	Berjalan, memukul
	Cara membunuh	Pemain menimpa dari atas musuh
	Nyawa	2
	Skor	10

Tabel 3. 14 Sketsa Karakter Musuh Ketiga

Sketsa	Profil	
	Nama	Buto Ijo
	Tampilan	Menyeramkan, berwarna hijau, berambut hijau panjang, membawa tongkat.
	Ukuran	Besar
	Kecepatan	Sedang
	Senjata	Tongkat
	Keahlian	Berjalan, memukul
	Cara membunuh	Pemain menimpa dari atas musuh
	Nyawa	9
	Skor	10

14. Matrik

Matrik menunjukkan hubungan ukuran karakter pemain dalam dunia permainan. Rancangan gambar matrik ditunjukkan pada Gambar 3.10.

Gambar 3. 10 Matrik

15. Koleksi

Selama permainan terdapat dua item yang harus di kumpulkan oleh pemain, yaitu jumlah potongan *puzzle* dan nyawa. Rancanganan ikon potongan *puzzle* dan ikon nyaawa di tunjukkan pada Gambar 3.11.

Gambar 3. 11 Koleksi

16. Nyawa

Pemain diberi 50 nyawa setiap kali bermain. Jika pemain terserang musuh pertama, pemain akan kehilangan 1 nyawa, terserang musuh kedua kehilangan 2 nyawa dan terserang pemain ke 3 kehilangan 3 nyawa. Apabila pemain berhasil mengambil bonus nyawa, nyawa pemain akan bertambah 10. Sketsa item nyawa ditunjukkan pada Gambar 3.11. Sedangkan posisi bar nyawa dalam dunia permaian ditunjukkan pada Tabel 3.7.

17. *Level*

Rancangan level pada *game* dijelaskan pada Tabel 3.15.

Tabel 3. 15 Rancangan Level pada *Game*

Level	Nama	Dunia Batik	Musuh
1	Pekalongan	Mengambil ciri batik pekalongan yang salah satunya berkesan cerah dan berwarna - warni dengan motif - motif yang sangat beragam.	Buto abang
2	Tasikmalaya	Mengambil ciri batik Tasikmalaya yaitu motif batik umumnya flora fauna dan menggunakan warna cerah.	Buto abang, buto kuning, jumlah semakin banyak.
3	Garut	Mengambil ciri batik Garut yaitu merupakan gambaran tentang kehidupan alam nyata.	Buto abang, Buto kuning
4	Solo	Mengambil ciri batik Solo yang umumnya memiliki warna coklat dan mengambil latar keraton yang berhubungan dengan Solo.	Buto abang, Buto kuning, Buto Ijo
5	Yogyakarta	Mengambil ciri batik Jogja yang umumnya memiliki warna coklat	Buto abang, Buto kuning, Buto ijo,

		bermotif mewah dan mengambil latar keraton yang berhubungan dengan Jogja.	jumlah semakin banyak.
6	Cirebon	Mengambil cirri batik Cirebon yaitu motif yang terkenal adalah mega mendung.	Buto abang yang super banyak.

18. Musik dan Suara

Musik yang digunakan dalam permaianan memiliki karakteristik cepat dan semangat agar membangkitkan semangat pemain.

3.2.2 *Technical Design*

Technical design menjelaskan aspek teknis dari pengembangan *game*. Yang meliputi pembuatan diagram *use case*, diagram *class* dan diagram *activity*.

3.2.2.1 *Use Case*

Use case merekam kebutuhan dari sebuah sistem. *Use case* dimodelkan dengan sebuah diagram use case. Pembuatan diagram *use case* dimulai dengan identifikasi aktor dan identifikasi kebutuhan.

1. Indentifikasi aktor

Tahap ini adalah tahap untuk melakukan identifikasi terhadap aktor yang akan berinteraksi dengan *game* Pengenalan Budaya Batik Indonesia. Tabel 3.16 memperlihatkan aktor yang terlibat beserta penjelasannya yang merupakan hasil dari proses identifikasi aktor.

Tabel 3. 16 Identifikasi aktor

Aktor	Deskripsi
Pemain	Pemain adalah pengguna yang dapat memainkan permaianan Batik Mania, mengontrol pemain, mendapatkan akses untuk melihat galeri, mengatur suara, melihat credit, dan melihat bantuan.

2. Daftar Kebutuhan

Daftar kebutuhan adalah daftar yang menjabarkan kebutuhan fungsional yang ada di dalam *permainan* Pengenalan Budaya Batik Indonesia, ditunjukkan pada Tabel 3.17.

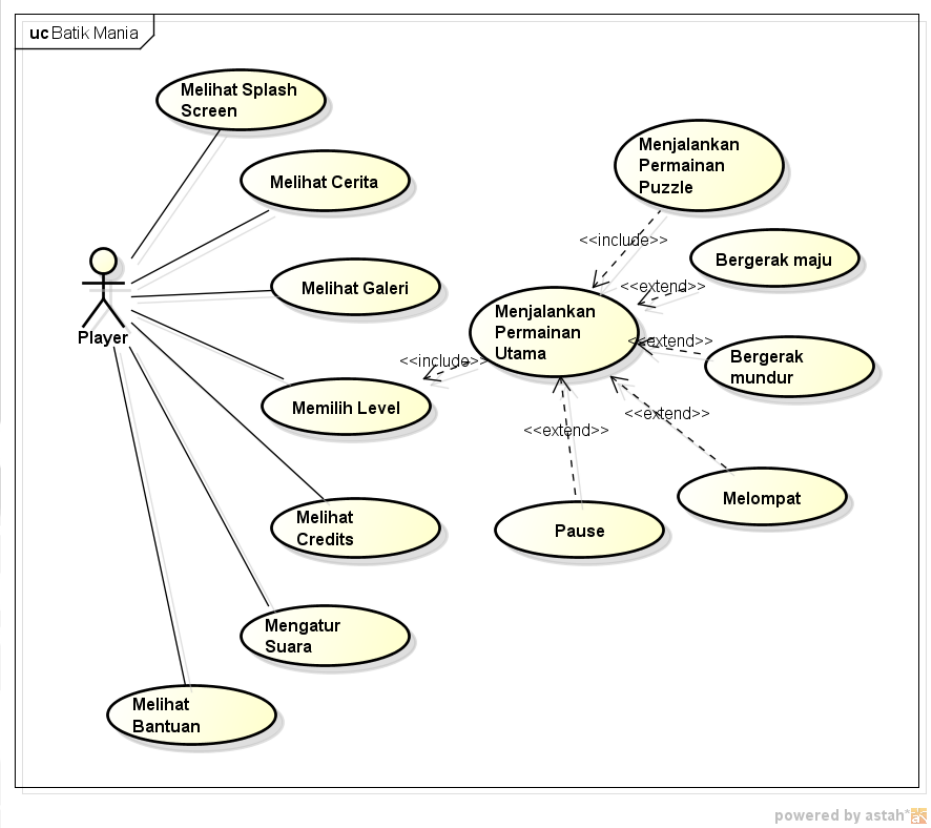
Tabel 3. 17 Spesifikasi Kebutuhan Fungsional Permainan

No	Kebutuhan	Use Case
1	*Game* harus dapat menampilkan *splash screen* logo UB, PTIIK dan GameLab sebelum masuk kedalam permainan.	Melihat Splash Screen
2	*Game* harus dapat menampilkan cerita permaian yang berbentuk komik sebanyak dua halaman yang disajikan dengan cara sentuh geser.	Melihat Cerita
3	*Game* harus dapat menyediakan fasilitas untuk meliahat deskripsi batik sesuai level permaian yang terbuka.	Melihat Galeri
4	*Game* harus menyediakan fasilitas untuk memilih level permainan. Level permainan berdasarkan nama batik yang akan diperkenalkan.	Memilih level
5	*Game* harus mampu memulai permainan dengan membuka layar untuk bermain saat pemain menekan tombol play.	Menjalankan permainan utama
6	*Game* harus mampu menggerakkan pemain ke depan saat pemain menekan tombol untuk maju.	Bergerak maju
7	*Game* harus mampu menggerakkan pemain ke belakang saat pemain menekan tombol untuk mundur.	Bergerak mundur
8	*Game* harus mampu menggerakkan pemain ke atas saat pemain menekan tombol untuk melompat.	Melompat
9	*Game* harus mampu memulai permainan *puzzle* dengan membuka layar untuk bermain ketika pemain berhasil menungumpulkan potongan *puzzle* pada permainan utama.	Menjalankan Permainan *puzzle*
10	*Game* harus menyediakan fasilitas untuk *pause game*. Sehingga pemain dapat menghentikan *game* untuk sementara waktu.	Pause Game
11	*Game* harus mampu menyediakan fasilitas pengaturan suara, apakah pemain ingin bermain dengan dengan menggunakan *sound effect* atau	Pengaturan suara

	tidak.	
12	*Game* harus menyediakan fasilitas untuk melihat *credit*. Sehingga pemain dapat melihat siapa saja yang terlibat dalam pembuatan *game*, referensi pembuatan *game* dan kontak pembuat *game*.	Melihat Credit
13	*Game* harus menyediakan fasilitas untuk melihat *bantuan*, yaitu halaman yang menunjukkan cara control permainan.	Melihat Bantuan

3. Diagram *Use Case*

Diagram *use case* digunakan untuk menggambarkan fungsionalitas dari *game* Pengenalan Budaya Batik Indonesia. Aktor dalam *use case* ini adalah pemain. Diagram *use* case dibuat berdasarkan identifikasi kebutuhan yang telah dilakukan sebelumnya. Model diagram *use case game* Batik Mania ditunjukkan pada Gambar 3.12.



Gambar 3. 12 Diagram *use case game* Pengenalan Budaya Batik Indonesia

3.2.2.2 Diagram *Class*

Diagram *class* memberikan gambaran pemodelan elemen-elemen *class* yang membentuk sebuah perangkat lunak. *Class* didapatkan dengan menganalisis secara detail terhadap *use case* yang dimodelkan. Diagram class pada *game* Pengenalan Budaya Batik Indonesia ditunjukkan pada Lampiran LM-1. Deskripsi dari *class game* Pengenalan Budaya Batik Indonesia ditunjukkan pada Tabel 3.18. Deskripsi dari *interface game* Pengenalan Budaya Batik Indonesia ditunjukkan pada Tabel 3.19.

Tabel 3. 18 Deskripsi diagram *class*

No	Class	Keterangan
1	`RMSGame`	Class yang berfungsi sebagai penyimpanan nilai skor.
2	`GameDesign`	Class yang menghandle art permainan.
3	`MenuDesign`	Class yang menghandle art story, main menu, gallery, credits, dan help.
4	`Main`	Class yang menghandle semua state dalam permainan, sehingga hanya perlu dipanggil satu kali dalam class main.
5	`Midlet`	Class yang berfungsi sebagai midlet permainan.
6	`AudioManager`	Class yang berfungsi untuk mengatur suara permaian.
7	`CreditsScreen`	Class yang berfungsi untuk mengatur screen credits.
8	`DeskripsiScreen`	Class yang berfungsi untuk mengatur screen deskripsi batik.
9	`GalleryScreen`	Class yang berfungsi untuk mengatur screen piling gallery.
10	`GameScreen`	Class yang berfungsi untuk mengatur screen permaian serta logika permainan.
11	`HelpScreen`	Class yang berfungsi untuk mengatur screen help.
12	`LevelScreen`	Class yang berfungsi untuk mengatur screen pilih level.
13	`MenuScreen`	Class yang berfungsi untuk mengatur screen main menu.
14	*`Puzzle`*`Screen`	Class yang berfungsi untuk mengatur screen *puzzle* serta logika permainan *puzzle*.
15	`SplashScreen`	Class yang berfungsi untuk mengatur screen splash.
16	`StoryScreen`	Class yang berfungsi untuk mengatur screen story.

17	`Dialog`	Class yang berfungsi untuk mengatur dialog permainan, seperti pause, reload dan reset. Yang mana nantinya dapat dipanggil oleh class – class screen.
18	`Enemy`	Class yang mengeksten class Sprite dan berfungsi mengatur method – method enemy.
19	`EnemyFactory`	Class yang memberikan masukan method – method yang telah di inisialisasi class enemy.
20	`EnemyPool`	Class yang mengatur munculnya enemy, yang bias di panggil oleh class GameScreen.
21	`Item`	Class yang mengeksten class Sprite dan berfungsi mengatur method – method item nyawa dan *puzzle*.
22	`ItemFactory`	Class yang memberikan masukan method – method yang telah di inisialisasi class item.
23	`ItemPool`	Class yang mengatur munculnya item nyawa dan *puzzle*, yang bisa di panggil oleh class GameScreen.
24	`PictFont`	Class yang mengatur pembuatan font – font secara manual dari image.
25	`Player`	Class yang mengatur logika pemain.
26	*`Puzzle`*	Class yang mengatur pemecahan gambar *puzzle* yang besar menjadi tile – tile kemudian mengacaknya.
27	*`Puzzle`*`Timer`	Class yang mengatur timer permainan *puzzle*.
28	`Tile`*`Puzzle`*	Class yang menghandle tile *puzzle* yang telah dipecah.

Tabel 3. 19 Deskripsi *Interface*

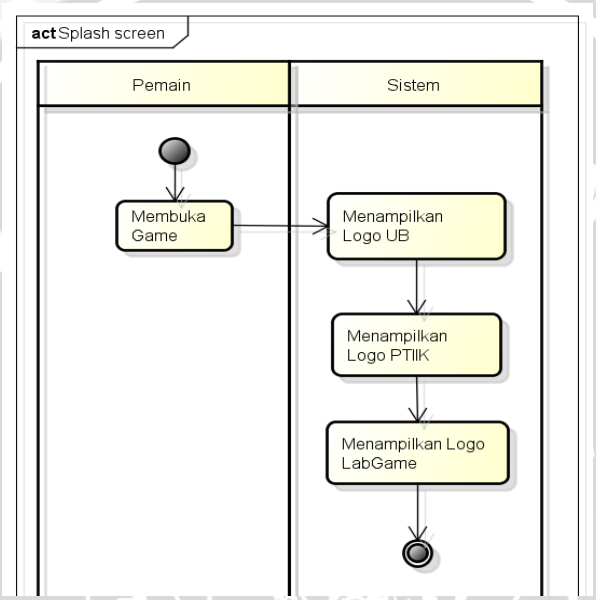
No	*Interface*	Keterangan
1	`StateInterface`	Interface yang berfungsi untuk diimplementasikan pada semua class screen.

3.2.2.3 Diagram *Activity*

Diagram *activity* digunakan untuk menjelaskan urutan dari aktivitas. Diagram *activity* menggambarkan alur kerja mulai dari *starting point* hingga *finish point* dengan memberikan detail alur percabangan yang terdapat pada proses aktivitas.

1. Diagram *Activity* Melihat *Splash Screen*

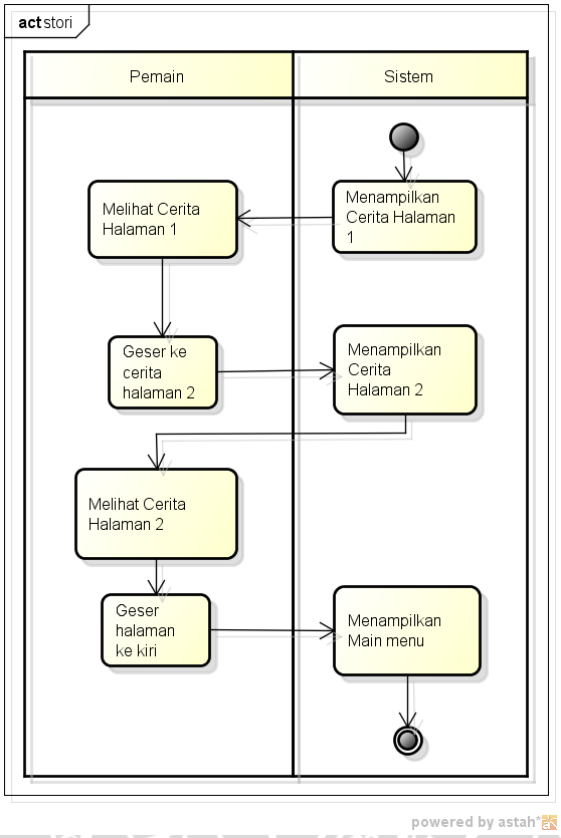
Diagram *activity* melihat *Splash Screen* dimulai dengan sistem menampilkan logo UB kemudia logo PTIIK dan terakhir logo LabGame secara berurutan. Diagram *activity* melihat *Splash Screen* ditunjukkan pada Gambar 3.13.



Gambar 3. 13 Diagram *Activity* Melihat *Splash Screen*

2. Diagram *Activity* Melihat Cerita

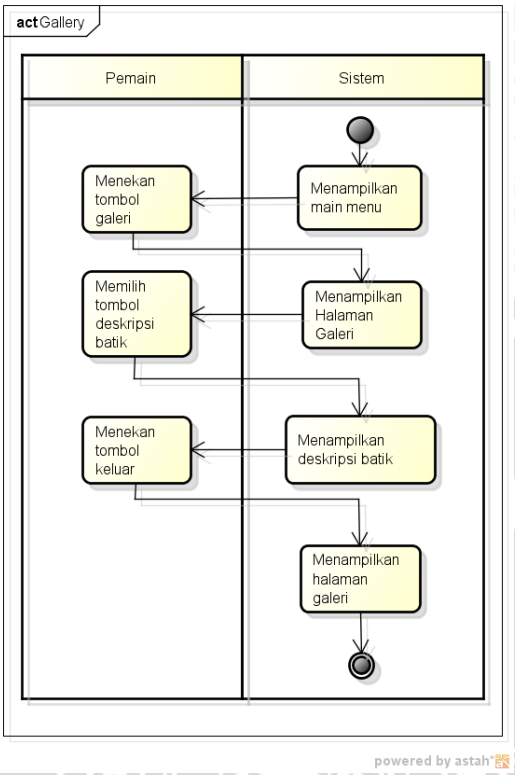
Diagram *activity* melihat cerita dimulai dengan sistem menampilkan halaman cerita pertama. Ketika pemain melakukan proses sentuh geser ke kanan sistem akan menampilkan cerita halaman kedua. Saat berada di halaman cerita kedua apabila pemain melakukan sentuh geser kekiri sistem akan kembali memunculkan halaman kesatu, sedangkan tekan geser ke kanan pemain akan masuk kedalam main menu. Diagram *activity* melihat cerita ditunjukkan pada Gambar 3.14.



Gambar 3. 14 Diagram *Activity* Melihat Cerita

3. Diagram *Activity* Melihat Galeri

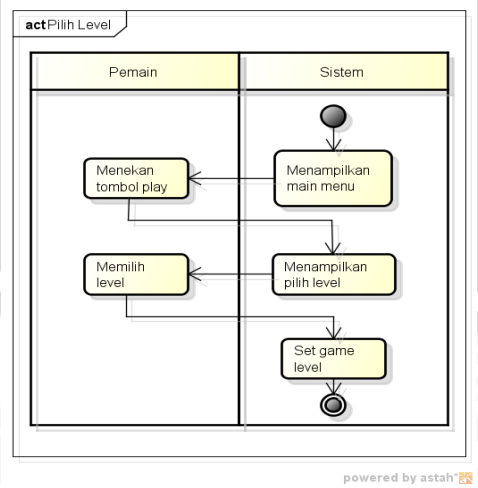
Diagram *activity* melihat galeri dimulai dengan pemain masuk ke dalam state galeri. Galeri berisi deskripsi – deskripsi batik yang telah di kumpulkan pemain. Pertama sistem akan menampilkan pilihan batik mana yang ingin dibuka. Kemudian sistem menampilkan deskripsi tentang detail batik secara keseluruhan yang dipilih pemain. Diagram *activity* melihat galeri ditunjukkan pada Gambar 3.15.



Gambar 3. 15 Diagram *Activity* Melihat Galeri

4. Diagram *Activity* Memilih *Level*

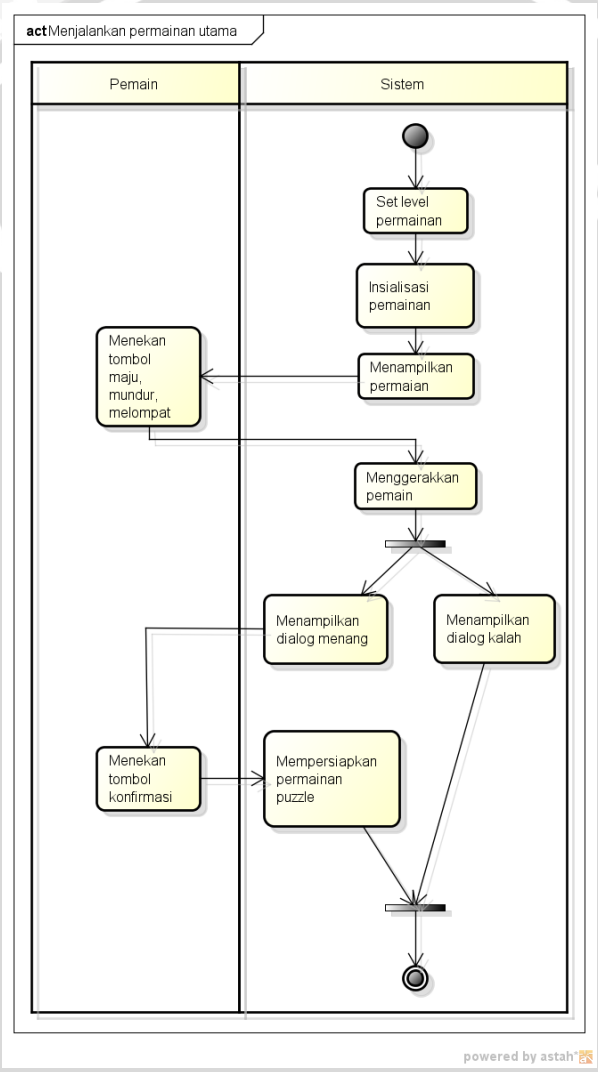
Diagram *activity* memilih *level* dimulai dengan pemain masuk ke dalam state memilih *level*. Sistem akan menampilkan pilihan *level* yang bisa dipilih pemain. Diagram *activity* memillih *level* ditunjukkan pada Gambar 3.16.



Gambar 3. 16 Diagram *Activity* Memilih *Level*

5. Diagram *Activity* Menjalankan Permainan Utama

Diagram *activity* menjalankan permainan utama adalah aktivitas untuk mulai bermain dalam permainan utama. Aktivitas dimulai oleh pemain dengan melakukan aktivitas memilih level. Setelah itu *game* menampilkan *scene* persiapan untuk mulai permainan sesuai level yang dipilih. Diagram *activity* menjalankan permainan utama ditunjukkan pada Gambar 3.17.

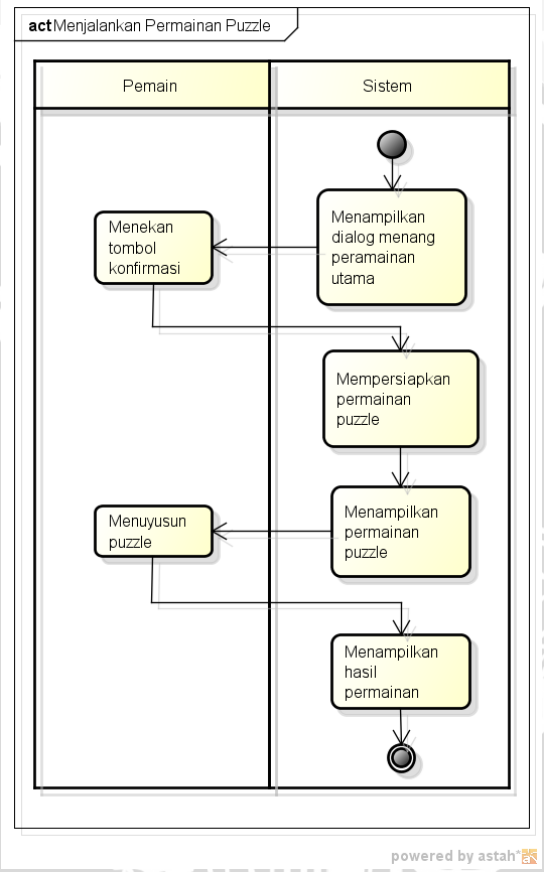


Gambar 3. 17 Diagram *Activity* Menjalankan Permainan Utama

6. Diagram *Activity* Menjalankan *Game Puzzle*

Diagram *activity* menjalankan *game puzzle* adalah aktivitas untuk mulai bermain dalam permainan *puzzle*. Aktivitas dimulai oleh pemain dengan melakukan menyelesaikan aktivitas menyelesaikan misi pada permainan utama.

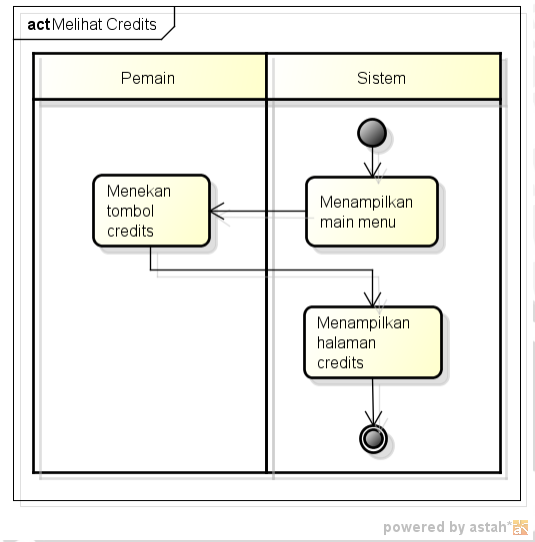
Setelah itu *game* menampilkan *scene* pemainan *puzzle* sesuai level yang dijalankan pada permainan utama. Diagram *activity* menjalankan permainan *puzzle* ditunjukkan pada Gambar 3.18.



Gambar 3. 18 Diagram *Activity* Menjalankan Permainan *Puzzle*

7. Diagram *Activity* Melihat Info

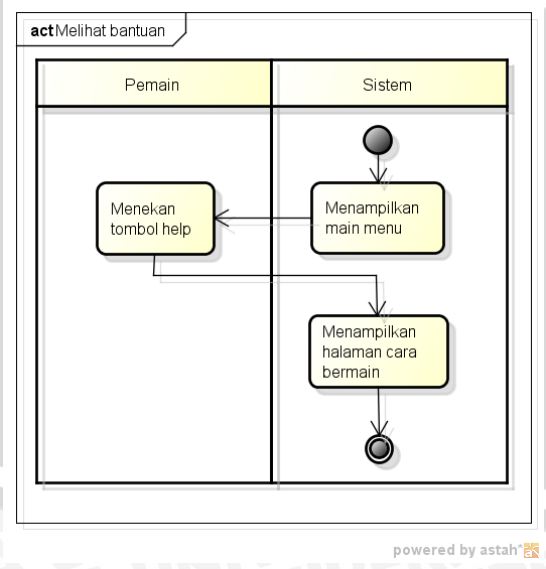
Diagram *activity* melihat info dimulai dengan pemain masuk ke dalam *state* info pembuat permaian. Info berisi tentang semua hal yangn berhubungan dengan pembuat permainan dan sumber-sumber terkait. Diagram *activity* melihat galeri ditunjukkan pada Gambar 3.19.



Gambar 3. 19 Diagram *Activity* Melihat Credits

8. Diagram *Activity* Melihat Bantuan

Diagram *activity* melihat bantuan dimulai dengan pemain masuk ke dalam *state* melihat bantuan. Bantuan berisi tentang cara bermain. Diagram *activity* melihat bantuan ditunjukkan pada Gambar 3.20.



Gambar 3. 20 Diagram *Activity* Melihat Bantuan

3.3 Implementasi *Game*

Implementasi pembangungan *game* dilakukan dengan mengacu kepada perancangan *game*. Implementasi pembuatan user interface dan karakter 2D menggunakan aplikasi Adobe Photoshop CS5, Adobe Ilustrator CS5 dan Adobe Flash CS5, sedangkan pembangunan perangkat lunak *game* menggunakan bahasa pemrograman berbasis objek *Java 2 Micro Edition* dengan *IDE Oracle Netbeans* 7.1 dan SDK 2.0.

3.4 Pengujian *Game*

Pengujian sistem merupakan hal penting yang bertujuan untuk menemukan kesalahan-kesalahan atau kekurangan-kekurangan pada perangkat lunak yang diuji serta melakukan pengujian kepada pengguna untuk memastikan perancangan game telah sesuai dengan tujuan yang ingin dicapai. Metode pengujian yang akan digunakan adalah *white-box testing*, *black-box testing*, pengujian *frame per second* (FPS) dan pengujian kepada pengguna dengan kuesioner. Pengujian bermaksud untuk mengetahui perangkat lunak yang dibuat sudah memenuhi kriteria yang sesuai dengan tujuan perancangan perangkat lunak tersebut.

BAB IV
IMPLEMENTASI

Bab ini membahas mengenai tahapan implementasi *game* Pengenalan Budaya Batik berdasarkan hasil yang telah didapatkan dari analisis kebutuhan dan proses perancangan *game*.

4.1 Spesifikasi Sistem

Perangkat lunak ini dikembangkan dalam lingkungan implementasi yang terdiri dari perangkat keras dan perangkat lunak.

4.1.1 Spesifikasi Lingkungan Perangkat Keras

Spesifikasi perangkat keras computer yang dipakai dalam proses pembangunan dijelaskan pada Tabel 4.1.

Tabel 4. 1 Spesifikasi Lingkungan Perangkat Keras Komputer

Perangkat Keras	**Spesifikasi**
Processor	Intel(R) Core(TM)2 Duo Processor T6600 (2.2 GHz, 800 MHz FSB, 2 MB L2 chace)
Memory (RAM)	2 GB
Harddisk	320 GB HDD
Motherboard	Acer Aspire 4736 Motherboard
Monitor	14” HD Acer CineCrystal (TM)

4.1.2 Spesifikasi Lingkungan Perangkat Lunak

Spesifikasi perangkat lunak yang dipakai dalam proses pengembangan perangkat lunak dijelaskan pada Tabel 4.2.

Tabel 4. 2 Spesifikasi Lingkungan Perangkat Lunak Komputer

Perangkat Lunak	**Spesifikasi**
Operating Sistem	Microsoft Windows 7 Ultimate 32-bit
Integrated Development Environment	Oracle NetBeans IDE 7.1
Programming Language	*Java*
Software Development Kit	SDK Nokia 2.0
Graphic Editor	Adobe Photoshop CS 5, Adobe Flash CS5 Adobe Illustrator CS 5

4.2 Batasan-Batasan Dalam Implementasi

Beberapa batasan dalam mengimplementasikan perangkat lunak ini adalah sebagai berikut:

1. *Game* single player.
2. *Game* menggunakan bahasa Indonesia.
3. Pengenalan batik yang di sajikan adalah Batik Pekalongan, Tasikmalaya, Garut, Solo, Yogyakarta dan Cirebon.

4.3 Implementasi *Art*

Implementasi *art* terdiri dari implementasi karakter dan implementasi lingkungan *game*.

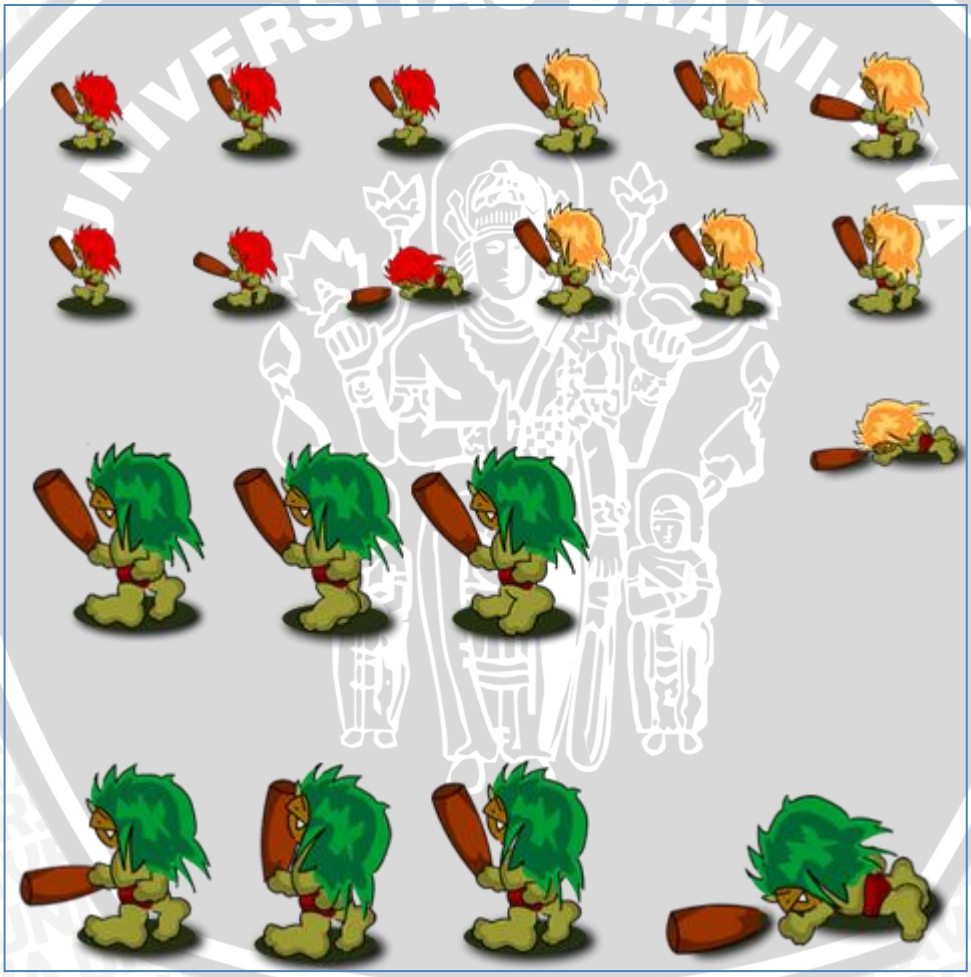
4.3.1 Implementasi *Art* Karakter

Implementasi sprite karakter menggunakan aplikasi *grafis editor*. Hasil implementasi karakter dalam *game* ditunjukkan pada Gambar 4.1.

Gambar 4. 1 Art Sprite Karakter

4.3.2 Implementasi *Art* Musuh

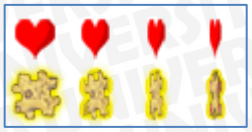
Implementasi sprite karakter musuh menggunakan aplikasi grafis editor. Hasil implementasi sprite musuh dalam *game* ditunjukkan pada Gambar 4.2.



Gambar 4. 2 *Art Sprite* Musuh

4.3.3 Implementasi *Art* Item Koleksi

Implementasi *sprite* item koleksi *puzzle* dan item koleksi nyawa ditunjukkan pada Gambar 4.3.



Gambar 4. 3 *Art Sprite* Item Koleksi

4.3.4 Implementasi *Art* Dunia Dalam Permainan

Pembuatan *art* 2D dunia permainan di implementasikan dalam bentuk kesatuan gambar yang nantinya akan dipotong membentuk tile-tile dan disusun membentuk dunia permainan. Gambar dikelompokkan sesuai *level* dan kebutuhan, agar saat program memuat gambar tidak menghabiskan memori yang besar. Gambar untuk tile level satu ditunjukkan pada Gambar 4.4. Gambar untuk tile level 2 ditunjukkan pada Gambar 4.5. Gambar untuk tile level 3 ditunjukkan pada Gambar 4.6. Gambar untuk tile level 4 ditunjukkan pada Gambar 4.7. Gambar untuk tile level 4 ditunjukkan pada Gambar 4.8. dan Gambar untuk tile level 6 ditunjukkan pada Gambar 4.9. Terdapat beberapa gambar tile yang dipakai oleh semua level ditunjukkan pada Gambar 4.10.

Gambar 4. 4 *Art* Tile Level Satu

Gambar 4. 5 *Art* Tile Level Dua

Gambar 4. 6 *Art* Tile Level Tiga

Gambar 4. 7 *Art* Tile Level Empat

Gambar 4. 8 *Art* Tile Level Lima

Gambar 4. 9 *Art* Tile Level Enam

Gambar 4. 10 *Art* Tile Semua Level

4.3.5 Implementasi *Art Puzzle*

Implementasi gambar *puzzle* yang akan dipotong menjadi ptotongan-potongan *puzzle* untuk disusun dalam pemain *puzzle* ditunjukkan pada Tabel 4.3. Implementasi merupakan gambar orang yang mengenakan pakaian batik sesuai level yang dimainkan.

Tabel 4. 3 Gambar *Art Puzzle*

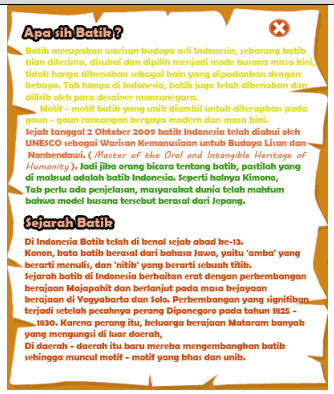
Level	Gambar
1. Pekalongan	
2. Tasikmalaya	
3. Garut	

4. Solo	
5. Yogyakarta	
6. Cirebon	

4.3.6 Implementasi *Art* Deskripsi Batik

Implementasi *art* deskripsi batik masing-masing level ditunjukkan pada Tabel 4.4.

Tabel 4. 4 *Art* Deskripsi Batik

Level	Gambar
Sejarah	
1. Pekalongan	
2. Tasikmalaya	
3. Garut	

4. Solo	
5. Yogyakarta	
6. Cirebon	

4.4 Implementasi Prosedur Program

Game Batik Mania memiliki beberapa proses atau *method* yang terdapat pada beberapa *class*. Pada penulisan skripsi ini hanya dicantumkan prosedur dari beberapa proses (operasi) utama saja sehingga tidak semua *method* akan dicantumkan. Implementasi prosedur ini akan direpresentasikan dalam bentuk *pseudocode*.

4.4.1 Prosedur Pegerakan Pemain

Prosedur Pergerakan Pemain merupakan proses yang akan menjelaskan langkah-langkah pergerakan pemain, mulai dari berlari, melompat dam diam. Implementasi Prosedur Pergerakan Pemain ditunjukkan pada tabel 4.5.

Tabel 4. 5 Implementasi Prosedur Pergerakan Pemain

***Pseudocode* Pergerakan Pemain**
Method: moveRight **DECLARATION:** moveRight, onGround IS Boolean **DESCRIPTION:** 9. moveRight <- TRUE 10. IF (onGround) THEN 11. CALL player.setFrameSequence(playerWalk) 12. CALL player.setTransform(0) 13. END IF 14. END moveRight
Method: moveLeft **DECLARATION:** moveLeft, onGround IS Boolean **DESCRIPTION:** 1. moveLeft <- TRUE 2. IF (onGround) THEN 3. CALL player.setFrameSequence(playerWalk) 4. CALL player.setTransform(2) 5. END IF 6. END moveLeft
Method: jump **DECLARATION:** moveLeft, onGround IS Boolean velocity IS Float

```
    maxJump IS Int
DESCRIPTION:
    1.  CALL player.setFrameSequence(playerJump)
    2.  IF (onGround) THEN
    3.        velocityY <- (-maxJump)
    4.        onGround <- FALSE
    5.        fromJump <- TRUE
    6.  END IF
    7.  END jump
```

Method: movement

```
DECLARATION:
    moveRight, moveLeft, reachLeft, reachLeft, fromJump, isStand,
    onGround IS Boolean
    Possx, Possy, maxRight, maxLeft, Speed, PossTanah IS Int
    velocity, gravity IS Float

DESCRIPTION:
    1.  IF (moveRight) THEN
    2.      IF (Possx < maxRight) THEN
    3.          Possx <- Possx + Speed
    4.          reachRight <- FALSE
    5.      ELSE
    6.          reachRight <- TRUE
    7.      END IF
    8.  END IF
    9.
    10. IF (moveLeft) THEN
    11.     IF (Possx > maxLeft)THEN
    12.         Possx <- Possx - Speed
    13.         reachLeft <- FALSE
    14.     ELSE
    15.         reachLeft <- TRUE
    16.     END IF
    17. END IF
    18.
    19. velocityY <- velocityY + gravity
    20. Possy <- Possy + velocity
    21.
    22. IF (Possy > PossTanah) THEN
```

```
23.      Possy <- PossTanah
24.      velocityY <- 0.0
25.      onGround <- true
26.      IF (fromJump)THEN
27.          IF (moveRight)THEN
28.              CALL player.setFrameSequence(playerWalk)
29.          ELSE
30.              CALL player.setFrameSequence(playerIdle)
31.              fromJump <- false
32.          END IF
33.      END IF
34.  END IF
35.
36.  IF (!moveRight && !isStand && onGround) THEN
37.      CALL player.setFrameSequence(playerIdle)
38.      isStand <- TRUE
39.  END IF
40.
41.  CALL player.nextFrame()
42.  CALL player.setPosition(Possx, Possy)
43.  END movement
```

Penjelasan dari Tabel 4.5 sebagai berikut :

Kolom pertama beisi method `moveRight()`, method ini berfungsi untuk menangani kondisi pemain bergerak maju. Ketika terjadi pemanggilan method ini maka moveRight ditandai true, kemudian dilakukan pengecekan ketika pemain onGround maka Frame Sequence pemain di set playerwalk.

Kolom kedua beisi method `moveLeft()`, method ini berfungsi untuk menangani kondisi pemain bergerak mundur. Ketika terjadi pemanggilan method ini maka moveLeft ditandai true, kemudian dilakukan pengecekan ketika pemain onGround maka Frame Sequence pemain di set playerwalk dengan posisi pemain di rotasi 180.

Kolom ketiga beisi method `jump()`, method ini berfungsi untuk menangani kondisi pemain bergerak melompat. Ketika terjadi pemanggilan method ini maka Frame Sequence di set jump, kemudian dilakukan pengecekan

ketika pemain onGround maka variable velocity diisi dengan decrement maxJump. Dan pemain ditandai tidak berada dalam posisi onGround tetapi fromJump.

Kolom ketiga berisi method `movement()` yaitu, Baris 1 - 8 merupakan pengecekan apakah pemain bergerak kedepan. Pertama pengecekan apakan ada inputan moveRight. Jika ada program melakukan pengecekan kembali. Jika posisi X pemain kurang dari batas kanan maksimal yang telah ditentukan maka pemain posisi X pemain ditambah kecepatan pemain, sehingga seolah olah pemain bergerak maju.

Baris 10 - 17 merupakan pengecekan apakah pemain bergerak kedepan. Pertama pengecekan apakan ada inputan moveLeft. Jika ada program melakukan pengecekan kembali. Jika posisi X pemain lebih dari batas kiri maksimal yang telah ditentukan maka pemain posisi X pemain ditambah kecepatan pemain, sehingga seolah olah pemain bergerak kebelakang.

Baris 19 – 34 adalah bagian yang menangani gerakan melompat pemain. Pertama ditentukan dahulu kecepatan lompatan pada baris 19, untuk menentukan posisi Y pemain pada baris 20. Pada baris 22 – 34 merupakan pengecekan apakah pemain telah melakukan gerakan melompat. Gerakan melompat terdapat pada method `jump()` yaitu ditentukkan maksimal tinggi lompatan pemain kemudian di decrement. Pertama pengecekan apakan posisi Y pemain lebih besar dari posisi Y tanah yang ditentukan. Jika iya posisi Y pemain dibuat sama sengan posisi Y tanah. Kemudian melakukan pengecekan lagi. Jika keadaan pemain moveRight maka FrameSequence di set walk, jika tidak diset idle. Sehingga seolah – olah pemain melompat dan sampai ke tanah.

Baris 36 – 39 merupakan pengecekan apakah pemain diam. Jika ya maka FrameSquence diset idle. Baris 41 merupakan pemanggilan method untuk memanggil frame selanjutnya dalam sprite. Baris 42 merupakan pemanggilan method untuk memberikan masukan update posisi pemain.

4.4.2 Prosedur Interaksi Musuh

Prosedur Interaksi Musuh merupakan proses yang akan menjelaskan langkah-langkah interaksi musuh dalam permainan. Implementasi Prosedur Interaksi Musuh ditunjukkan pada tabel 4.6.

Tabel 4. 6 Implementasi Prosedur Interaksi Musuh

***Pseudocode* Interaksi Musuh**
Method: hit **DECLARATION:** Hitted, isDeath IS Boolean power, score, deathpos IS Int deadTimer IS Long **DESCRIPTION:** 1. Hitted <- true 2. Power <- Power - 1 3. GameScreen.score <- GameScreen.score + 10 4. IF (power <= 0) THEN 5. CALL this.setFrameSequence(enemyDie) 6. deadTimer <- Sistem.currentTimeMillis() 7. isDeath <- true 8. deathpos <- super.getX() 9. END IF 10. END hit
Method: move PARAMETER feedback, value **DECLARATION:** feedback IS Boolean value, speedAll, speed IS Int **DESCRIPTION:** 1. IF (Enemy <- Visible) THEN 2. IF (feedback <- TRUE) THEN 3. speedAll <- speed + value 4. ELSE 5. speedAll <- speed 6. IF (!collidesWith(player))THEN 7. CALL super.move(-speedAll, 0) 8. ELSE 9. CALL hit() 10. IF (attacking) THEN 11. CALL player.hit(attackPower) 12. IF (isDeath) THEN 13. CALL setVisible(false) 14. END IF 15. END IF 16. END move

```
Method: cycle

DECLARATION:
    feedback IS Boolean
    value, speedAll, speed IS Int

DESCRIPTION:
  1.  FOR (int i=0 TO i<enemies.size()THEN i++)THEN
  2.        CALL ((Enemy)enemies.elementAt(i)).move(feedback, value)
  3.  END FOR
  4.  END cycle
```

Penjelasan dari Tabel 4.6 sebagai berikut :

Kolom pertama berisi method `hit()` method ini merupakan logika ketika musuh mengalami serangan dari pemain. Method ini ditandai dengan hitted di set true. Kemudian power musuh dikurangi, skor pemain ditambah, Frame Squence musuh di set die dan ditentukan posisinya X nya.

Kolom kedua berisi method `move()` denga pharameter feedback dan value, parameter ini berfungsi untuk memberikan masukan dan pengecekan apakah saat musuh berjalan kondisi pemain juga berjalan. Baris 1 – 5 merupakan penyetingan kecepatan musuh. Jika feedback ditandai dengan true maka terjadi umpan balik dari pemain, maka kecepatan jalan musuh sama dengan kecepatan awal ditambah dengan kecepatan masukan. Jika tidak terjadi umpan balik maka kecepatan musuh tetap. Baris 6 – 9 merupakan deteksi tabrakan musuh. Jika tidak terjadi tabrakan antara musuh dengan pemain makan musuh tetap berjalan dengan kecepatan yang telah ditentukan. Jika terjadi tabrakan maka terjadi pemanggilan method `hit()`.

Kolom ketiga berisi method cycle, mehod ini merupakan logika perulangan kedatangan musuh. Ketika jumlah i lebih besar dari jumlah musuh, maka terjadi pemanggilan musuh.

4.4.3 Prosedur Interaksi Item

Prosedur Interaksi Item merupakan proses yang akan menjelaskan langkah-langkah interaksi item potongan *puzzle* dan item nyawa dalam permainan. Implementasi Prosedur Interaksi Item ditunjukkan pada Tabel 4.7.

Tabel 4. 7 Implementasi Prosedur Interaksi Item

***Pseudocode* Interaksi Item**
Method: kena **DECLARATION:** isBatik, isNyawa IS Boolean speed IS Int **DESCRIPTION:** 1. IF (isBatik) THEN 2. GameScreen.*puzzle*piece <- GameScreen.*puzzle*piece + 1 3. IF (isNyawa) 4. CALL player.hit(-10) 5. END IF 6. END kena
Method: muncul **DECLARATION:** Boolean feedback, Int value, speedAll, speed **DESCRIPTION:** 1. IF (Item <- Visible) THEN 2. IF (!collidesWith(player))THEN 3. CALL super.move(-speed, 0) 4. ELSE 5. CALL kena() 6. setVisible <- false 7. END IF 8. END IF 9. END muncul
Method: cycle **DECLARATION:** feedback IS Boolean value, speedAll, speed IS Int **DESCRIPTION:** 1. FOR (int i=0 TO i< batikItems.size()THEN i++)THEN 2. CALL ((Enemy) batikItems.elementAt(i)).muncul() 3. END FOR 4. FOR (int i=0 TO i< healthItems.size()THEN i++)THEN 5. CALL ((Enemy) healthItems.elementAt(i)).muncul() 6. END FOR 7. END cycle

Penjelasan dari Tabel 4.7 sebagai berikut :

Kolom pertama berisi method `kena()`, method ini merupakan prosedur untuk proses ketika item telah bertabrakan dengan pemain. Baris ke 1 – 2 merupakan pengecekan kondisi item adalah potongan *puzzle*, maka jumlah piece *puzzle* bertambah. Baris ke 3 – 4 merupakan pengecekan kondisi item adalah nyawa, maka nyawa pemain bertambah 10.

Kolom kedua berisi berisi method `muncul()`, method ini merupakan prosedur pergerakan item dan pengecekan tabrakan dengan pemain. Baris 1 adalah pengecekan apakah item yang akan di cek masih ada. Baris 2 merupakan pengecekan apakah terjadi tabrakan item dengan pemain, jika tidak maka item terus berjalan melalui pemanggilan method `move(pharameter x,y)`, jika ya maka program akan mengeksekusi baris 4 – 6 yaitu pemanggilan method kena() kemudian item di set visible false.

Kolom ketiga berisi method `cycle()`, method ini merupakan prosedur perulangan kedatangan item piece *puzzle* dan nyawa. Ketika jumlah i lebih besar dari jumlah item, maka akan terjadi pemanggilan item.

4.4.4 Prosedur Pengecekan Permainan *Puzzle*

Prosedur Pengecekan Permainan *Puzzle* merupakan proses yang akan menjelaskan langkah-langkah pengecekan kondisi menang permaian *puzzle*. Implementasi Prosedur Interaksi Item ditunjukkan pada tabel 4.8.

Tabel 4. 8 Implementasi Prosedur Pengecekan *Puzzle*

Pseudocode* Pengecekan Permainan *Puzzle
Method: tileMatch **DECLARATION:** x, y IS Int **DESCRIPTION:** 1. IF (x = currentx && y = currenty) 2. Return <- true 3. END IF 4. Return <- false 5. END tileMatch

```
Method: winCheck

DECLARATION:
     grid IS Int

DESCRIPTION:
    1.   FOR (int i=0 TO i < grid THEN i++)
    2.         FOR (int j=0 TO j < grid THEN j++)
    3.               IF (!tiles[i][j].tileMatch())
    4.                     return FALSE
    5.         END FOR
    6.   END FOR
    7.   return TRUE
```

Penjelasan dari Prosedur Pengecekan kondisi menang pada Tabel 4.8 yaitu :

Kolom pertama berisi method `tileMatch()`, method ini berisi pengecekan ketika posisi x dan y potongan *puzzle* acak sesuai dengan posisi x dan y awal, maka dilakukan kembalian true, jika tidak maka dilakukan kembalian false.

Kolom kedua berisi method `winCheck()`, method ini adalah method yang berisi prosedur keseluruhan pengecekan kondisi menang pemain. Pertama – tama sistem melakukan pengecekan secara berulang - ulang terhadap index tile *puzzle*. Proses pengecekan yaitu memanggil method tileMatch yang bertipe Boolean. Apabila posisi x dan y piece *puzzle* yang telah teracak sama dengan x dan y awal, sistem akan memberi kembalian true yang berarti *puzzle* telah tersusun.

4.5 Implementasi Antarmuka dan HUD

Implementasi antarmuka dalam *game* Pengenalan Budaya Batik Indonesia meliputi implementasi antarmuka dan implementasi sistem *Head-Up Display* (HUD). Implementasi layar Layar Awal *game* ditunjukkan pada Gambar 4.11. Implementasi layar cerita ditunjukkan pada Gambar 4.12. Implementasi layar menu utama ditunjukkan pada Gambar 4.13. Implementasi layar pilih level ditunjukkan pada Gambar 4.14. Implementasi layar *gameplay* ditunjukkan pada Gambar 4.15 Implementasi layar *pause* permainan ditunjukkan pada Gambar 4.16. Implementasi layar *game puzzle* ditunjukkan pada Gambar 4.17.

Implementasi layar galeri ditunjukkan pada Gambar 4.18. Implementasi layar deskripsi batik ditunjukkan pada Gambar 4.19. Implementasi layar cara bermain ditunjukkan pada Gambar 4.20. Implementasi layar informasi permainan ditunjukkan pada Gambar 4.21. Implementasi HUD permainan utama ditunjukkan pada Gambar 4.22. Implementasi HUD permainan *puzzle* ditunjukkan pada Gambar 4.23.

Gambar 4. 11 Implementasi Antarmuka Layar Awal

Gambar 4. 12 Implementasi Antarmuka Cerita

Gambar 4. 13 Implementasi Antarmuka Menu Utama

Gambar 4. 14 Implementasi Antarmuka Pilih Level

Gambar 4. 15 Implementasi Antarmuka *Gameplay*

Gambar 4. 16 Implementasi Antarmuka Pause

Gambar 4. 17 Implementasi Antarmuka Permainan *Puzzle*

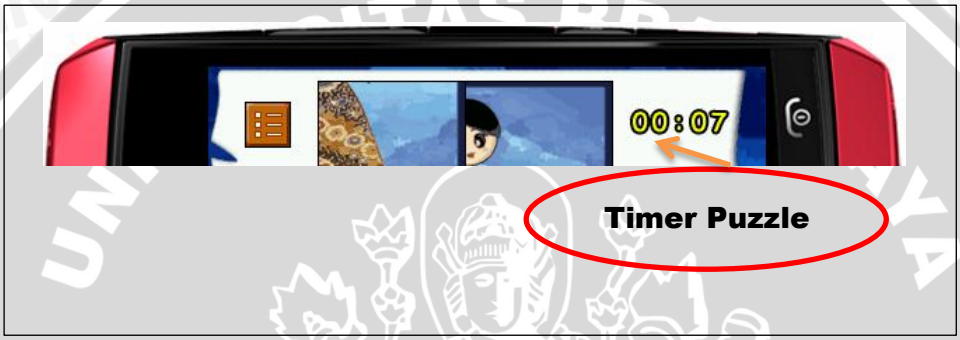
Gambar 4. 18 Implementasi Antarmuka Galeri

Gambar 4. 19 Implementasi Antarmuka Deskripsi Batik

Gambar 4. 20 Implementasi Antarmuka Cara Bermain

Gambar 4. 21 Implementasi Antarmuka Informasi Permainan

Gambar 4. 22 Implementasi HUD Permainan Utama



Gambar 4. 23 Implementasi HUD Permainan *Puzzle*

BAB V
PENGUJIAN DAN ANALISIS

Bab ini membahas mengenai tahapan pengujian dan analisis *game* Pengenalan Budaya Batik Indonesia yang telah dikembangkan. Proses pengujian dilakukan melalui empat tahapan yaitu pengujian unit, pengujian integrasi, pengujian validasi, dan pengujian performa. Pada pengujian unit dan integrasi, akan digunakan teknik pengujian *White-Box* (*White-Box Testing*). Pada pengujian validasi akan digunakan teknik pengujian *Black-Box* (*Black-Box Testing*). Pada pengujian performa, subjek yang di uji adalah nilai *frames per second*.

5.1 Pengujian

Proses pengujian yang dilakukan melalui empat tahapan yaitu pengujian unit, pengujian integrasi, pengujian validasi dan pengujian kinerja.

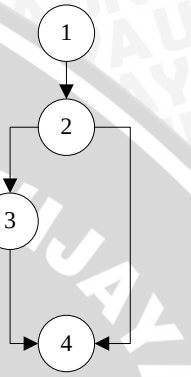
5.1.1 Pengujian Unit

Pada pengujian unit *game* Pengenalan Budaya Batik Indonesia ini, digunakan teknik pengujian *White-Box* dengan teknik *Basis Path Testing*. Pada teknik *Basis Path Testing*, proses pengujian dilakukan dengan memodelkan *Prosedur* pada suatu *flow graph*, menentukan jumlah kompleksitas siklomatis (*cyclomatic complexity*), menentukan sebuah basis set dari jalur independen dan memberikan kasus uji (*test case*) pada setiap basis set yang telah ditentukan. Penulisan laporan skripsi ini hanya dicantumkan hasil pengujian unit untuk *Prosedur* dari beberapa metode (operasi) saja (tidak untuk keseluruhan metode).

1. Pengujian Unit Operasi `moveRight()`

Operasi `moveRight()` merupakan operasi untuk prosedur pemain bergerak kedapan, yang diimplementasikan dalam method `moveRight()`. Pemodelan operasi `moveRight()` dalam bentuk *flow graph* ditunjukkan pada Tabel 5.1.

Tabel 5. 1 Pemodelan *Flow Graph* operasi `moveRight()`

Pseudocode	***Flow Graph***
Method: moveRight **DECLARATION:** moveRight, onGround IS Boolean **DESCRIPTION:** 1. moveRight <- TRUE 2. IF (onGround) THEN 3. CALL player.setFrameSequence(playerWalk) 4. CALL player.setTransform(0) 5. END IF 6. END moveRight	Node (N) = 4 Edge (E) = 4 

Pemodelan ke dalam *flow graph* yang telah dilakukan terhadap operasi moveRight () menghasilkan jumlah kompleksitas siklomatis *(cyclomatic complexity)* melalui persamaan V(G) = E – N + 2 .

V(G) = E – N + 2

= 4 – 4 + 2

= 2

Berdasarkan dari nilai *cyclomatic complexity* yang telah didapatkan dari perhitungan maka ditentukan satu buah basis set dari jalur *independent*, yaitu :

Jalur 1: 1 – 2 – 4

Jalur 2: 1 – 2 – 3 – 4

Penentuan kasus uji untuk jalur *independent* tersebut dan hasil eksekusinya dijelaskan pada Tabel 5.2.

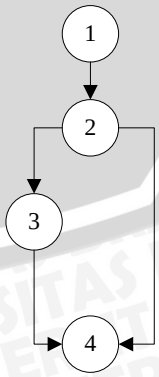
Tabel 5. 2 *Test Case* Pengujian operasi `moveRight()`

Jalur	Kasus Uji	Hasil yang diharapkan	Hasil yang didapatkan
1	Mengeksekusi perintah dengan onGround di set true.	Sistem memproses Frame Squence pemain walk dengan Transform none.	Sistem memproses Frame Squence pemain walk dengan Transform none.
2	Mengeksekusi perrintah dengan onGround di set false.	Sistem tidak memproses Frame Squence pemain.	Sistem tidak memproses Frame Squence pemain.

2. Pengujian Unit Operasi `moveLeft()`

Operasi `moveLeft()` merupakan operasi untuk prosedur pemain bergerak kebelakang, yang diimplementasikan dalam method `moveLeft()`. Pemodelan operasi `moveLeft()` dalam bentuk *flow graph* ditunjukkan pada Tabel 5.3.

Tabel 5. 3 Pemodelan *Flow Graph* operasi `moveLeft()`

Pseudocode	*Flow Graph*
Methode: moveLeft **DECLARATION:** moveLeft, onGround IS Boolean **DESCRIPTION:** 1. moveLeft <- TRUE 2. IF (onGround) THEN 3. CALL player.setFrameSequence(playerWalk) 4. CALL player.setTransform(2) 5. END IF 6. END moveLeft	Node (N) = 4 Edge (E) = 4 

Pemodelan ke dalam *flow graph* yang telah dilakukan terhadap operasi moveLeft() menghasilkan jumlah kompleksitas siklomatis *(cyclomatic complexity)* melalui persamaan V(G) = E – N + 2 .

V(G) = E – N + 2
= 4 – 4 + 2
= 2

Berdasarkan dari nilai *cyclomatic complexity* yang telah didapatkan dari perhitungan maka ditentukan satu buah basis set dari jalur *independent*, yaitu :

Jalur 1: 1 – 2 – 4

Jalur 2: 1 – 2 – 3 – 4

Penentuan kasus uji untuk jalur *independent* tersebut dan hasil eksekusinya dijelaskan pada Tabel 5.4.

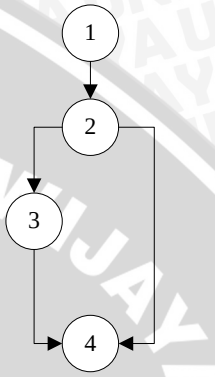
Tabel 5. 4 *Test Case* Pengujian operasi `moveLeft()`

Jalur	Kasus Uji	Hasil yang diharapkan	Hasil yang didapatkan
1	Mengeksekusi perintah dengan onGround di set true.	Sistem memproses Frame Squence pemain walk dengan Transform rotate 180.	Sistem memproses Frame Squence pemain walk dengan Transform rotate 180.
2	Mengeksekusi perintah dengan onGround di set false.	Sistem tidak memproses Frame Squence pemain dan Transform pemain tetap none.	Sistem tidak memproses Frame Squence pemain dan Transform pemain tetap none.

3. Pengujian Unit Operasi `jump()`

Operasi `jump()` merupakan operasi untuk prosedur pemain bergerak melakukan gerakan melompat, yang diimplementasikan dalam method `jump()`. Pemodelan operasi `jump()` dalam bentuk *flow graph* ditunjukkan pada Tabel 5.5.

Tabel 5. 5 Pemodelan *Flow Graph* operasi `jump()`

Pseudocode	***Flow Graph***
Methode: jump **DECLARATION:** moveLeft, onGround IS Boolean velocity IS Float maxJump IS Int **DESCRIPTION:** 1. CALL player.setFrameSequence(playerJump) 2. IF (onGround) THEN 3. velocityY <- (-maxJump) 4. onGround <- FALSE 5. fromJump <- TRUE 6. END IF 7. END jump	Node (N) = 4 Edge (E) = 4 

Pemodelan ke dalam *flow graph* yang telah dilakukan terhadap operasi jump() menghasilkan jumlah kompleksitas siklomatis *(cyclomatic complexity)* melalui persamaan V(G) = E – N + 2 .

V(G) = E – N + 2

= 4 – 4 + 2

= 2

Berdasarkan dari nilai *cyclomatic complexity* yang telah didapatkan dari perhitungan maka ditentukan satu buah basis set dari jalur *independent*, yaitu :

Jalur 1: 1 – 2 – 4

Jalur 2: 1 – 2 – 3 – 4

Penentuan kasus uji untuk jalur *independent* tersebut dan hasil eksekusinya dijelaskan pada Tabel 5.6.

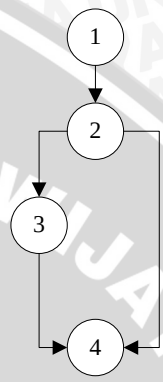
Tabel 5. 6 *Test Case* Pengujian operasi `jump()`

Jalur	Kasus Uji	Hasil yang diharapkan	Hasil yang didapatkan
1	Mengeksekusi perintah dengan onGround di set true.	VelocityY pemain sama dengan decrement maxJump, sehingga pemain terlihat bergerak keatas dan seolah olah melompat.	VelocityY pemain sama dengan decrement maxJump, sehingga pemain terlihat bergerak keatas dan seolah olah melompat.
2	Mengeksekusi perrintah dengan onGround di set false.	Sistem tidak mengeksekusi perintah yang ada di dalam kondisi if, sehingga posisi pemain ketika ditekan tombol jump tetap.	Sistem tidak mengeksekusi perintah yang ada di dalam kondisi if, sehingga posisi pemain ketika ditekan tombol jump tetap.

4. Pengujian Unit Operasi `hit()`

Operasi `hit()` merupakan operasi untuk prosedur musuh terserang pemain, yang diimplementasikan dalam method `hit()`. Pemodelan operasi `hit()` dalam bentuk *flow graph* ditunjukkan pada Tabel 5.7.

Tabel 5. 7 Pemodelan *Flow Graph* operasi `hit()`

Pseudocode	***Flow Graph***
Methode: hit **DECLARATION:** Hitted, isDeath IS Boolean power, score, deathpos IS Int deadTimer IS Long **DESCRIPTION:** 1. Hitted <- true 2. Power <- Power - 1 3. GameScreen.score <- GameScreen.score + 10 4. IF (power <= 0) THEN 5. CALL this.setFrameSequence(enemyDie) 6. deadTimer <- Sistem.currentTimeMillis() 7. isDeath <- true 8. deathpos <- super.getX() 9. END IF 10. END hit	Node (N) = 4 Edge (E) = 4 

Pemodelan ke dalam *flow graph* yang telah dilakukan terhadap operasi hit() menghasilkan jumlah kompleksitas siklomatis *(cyclomatic complexity)* melalui persamaan V(G) = E – N + 2 .

$$V(G) = E - N + 2$$
$$= 4 - 4 + 2$$
$$= 2$$

Berdasarkan dari nilai *cyclomatic complexity* yang telah didapatkan dari perhitungan maka ditentukan satu buah basis set dari jalur *independent*, yaitu :

Jalur 1: 1 – 2 – 4

Jalur 2: 1 – 2 – 3 – 4

Penentuan kasus uji untuk jalur *independent* tersebut dan hasil eksekusinya dijelaskan pada Tabel 5.8.

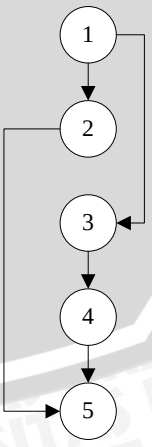
Tabel 5. 8 *Test Case* Pengujian operasi `hit()`

Jalur	Kasus Uji	Hasil yang diharapkan	Hasil yang didapatkan
1	Memberikan inputan power kurang dari 0	Program mengeksekusi enemy die.	Program mengeksekusi enemy die.
2	Memberikan inputan power lebih dari 0	Program tidak mengeksekusi enemy die.	Program tidak mengeksekusi enemy die.

5. Pengujian Unit Operasi `kena()`

Operasi `kena()` merupakan operasi untuk prosedur item berbenturan dengan pemain, yang diimplementasikan dalam method `kena()`. Pemodelan operasi `kena()` dalam bentuk *flow graph* ditunjukkan pada Tabel 5.9.

Tabel 5. 9 Pemodelan *Flow Graph* operasi `kena()`

Pseudocode	***Flow Graph***
Methode: kena **DECLARATION:** isBatik, isNyawa IS Boolean speed IS Int **DESCRIPTION:** 1. IF (isBatik) THEN 2. GameScreen.*puzzle*piece <- GameScreen.*puzzle*piece + 1 3. IF (isNyawa) 4. CALL player.hit(-10) 5. END IF 6. END kena	Node (N) = 5 Edge (E) = 5 

Pemodelan ke dalam *flow graph* yang telah dilakukan terhadap operasi kena() menghasilkan jumlah kompleksitas siklomatis *(cyclomatic complexity)* melalui persamaan V(G) = E – N + 2 .

V(G) = E – N + 2

= 5 – 5 + 2

= 2

Berdasarkan dari nilai *cyclomatic complexity* yang telah didapatkan dari perhitungan maka ditentukan satu buah basis set dari jalur *independent*, yaitu :

Jalur 1: 1 – 2 – 5

Jalur 2: 1 – 3 – 4 – 5

Penentuan kasus uji untuk jalur *independent* tersebut dan hasil eksekusinya dijelaskan pada Tabel 5.10.

Tabel 5. 10 *Test Case* Pengujian operasi `kena()`

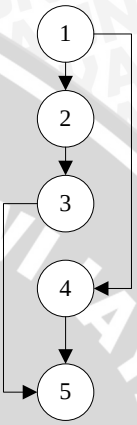
Jalur	Kasus Uji	Hasil yang diharapkan	Hasil yang didapatkan
1	Pengecekan kondisi isBatik true	Jumlah piece *puzzle* bertambah 1	Jumlah piece *puzzle* bertambah 1
2	Pengecekan kondisi isNyawa true	Jumlah nyawa pemain ditambah 10	Jumlah nyawa pemain ditambah 10

6. Pengujian Unit Operasi `tileMatch()`

Operasi `tileMatch()` merupakan operasi untuk prosedur pengecekan kondisi menang pada permainan *puzzle*, yang diimplementasikan dalam method `tileMatch()`. Pemodelan operasi `tileMatch()` dalam bentuk *flow graph* ditunjukkan pada Tabel 5.11.

Tabel 5. 11 Pemodelan *Flow Graph* operasi `tileMatch()`

Pseudocode	***Flow Graph***

`Methode: tileMatch`
 	
`DECLARATION:`
`x, y IS Int`
 	
`DESCRIPTION:`
`1. IF (x = currentx && y = currenty)`
`2. Return <-true`
`3. END IF`
`4. Return <- false`
`5. END tileMatch`
	`Node (N) = 5`
	`Edge (E) = 5`
	 

Pemodelan ke dalam *flow graph* yang telah dilakukan terhadap operasi jump () menghasilkan jumlah kompleksitas siklomatis *(cyclomatic complexity)* melalui persamaan V(G) = E – N + 2 .

V(G) = E – N + 2

= 5 – 5 + 2

= 2

Berdasarkan dari nilai *cyclomatic complexity* yang telah didapatkan dari perhitungan maka ditentukan satu buah basis set dari jalur *independent*, yaitu :

Jalur 1: 1 – 4 – 5

Jalur 2: 1 – 2 – 3 – 5

Penentuan kasus uji untuk jalur *independent* tersebut dan hasil eksekusinya dijelaskan pada Tabel 5.12.

Tabel 5. 12 *Test Case* Pengujian operasi `tileMatch()`

Jalur	Kasus Uji	Hasil yang diharapkan	Hasil yang didapatkan
1	Pengecekan kondisi nilai x dan y tidak sama dengan currentx dan currenty	Sistem melakukan kembalian false.	Sistem melakukan kembalian false.
2	Pengecekan kondisi nilai x dan y sama dengan currentx dan currenty	Sistem melakukan kembalian true.	Sistem melakukan kembalian true.

5.1.2 Pengujian Integrasi

Pengujian integrasi diterapkan pada proses yang mengintegrasikan fungsionalitas dari beberapa *class* untuk melakukan sebuah operasi tertentu. Pada pengujian integrasi yang dijadikan sebagai objek uji adalah *class* inti yang menggabungkan kinerja dari *class-class* lain. Pengujian integrasi yang dilakukan terhadap *class* inti ini menggunakan strategi *bottom-up,* dimana modul-modul yang diintegrasikan masing-masing diuji terlebih dahulu dalam pengujian unit dan kemudian bergerak menuju ke pengujian modul-modul kontrol yang mengintegrasikannya.

1. Pengujian Integrasi Operasi Pergerakan Pemain

Operasi pergerakan pemain merupakan operasi untuk prosedur pergerakan pemain pada permainan utama, yang diimplementasikan dalam method `movement()`. Pemodelan operasi `movement()` dalam bentuk *flow graph* ditunjukkan pada Tabel 5.13.

Tabel 5. 13 Pemodelan *Flow Graph* Operasi Pergerakan Pemain

Pseudocode	***Flow Graph***

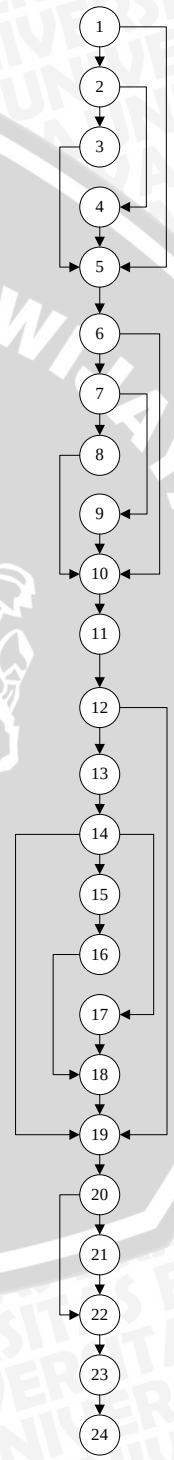
```
Methode: movement

DECLARATION:
    moveRight, moveLeft, reachLeft, reachLeft,
    fromJump, isStand, onGround IS Boolean
    Possx, Possy, maxRight, maxLeft, Speed,
    PossTanah IS Int
    velocity, gravity IS Float

DESCRIPTION:
1.  IF (moveRight) THEN
2.    IF (Possx < maxRight) THEN
3.      Possx <- Possx + Speed
4.      reachRight <- FALSE
5.    ELSE
6.      reachRight <- TRUE
7.
8.  END IF
9.  END IF
10.
11. IF (moveLeft) THEN
12.     IF (Possx > maxLeft)THEN
13.       Possx <- Possx - Speed
14.       reachLeft <- FALSE
15.     ELSE
16.       reachLeft <- TRUE
17.     END IF
18. END IF
19.
20. velocityY <- velocityY + gravity
21. Possy <- Possy + velocity
22.
23. IF (Possy > PossTanah) THEN
24.   Possy <- PossTanah
25.   velocityY <- 0.0
26.   onGround <- true
27.   IF (fromJump)THEN
```

Node (N) = 24

Edge (E) = 31



Pemodelan ke dalam *flow graph* yang telah dilakukan terhadap operasi pergerakan pemain menghasilkan jumlah kompleksitas siklomatis *(cyclomatic complexity)* melalui persamaan V(G) = E – N + 2 .

V(G) = E – N + 2

= 31 – 24 + 2

= 9

Berdasarkan dari nilai *cyclomatic complexity* yang telah didapatkan dari perhitungan maka ditentukan satu buah basis set dari jalur *independent*, yaitu :

Jalur 1 : 1-5-6-10-11-12-19-20-22-23-24

Jalur 2 : 1-2-4-5-6-10-11-12-19-20-22-23-24

Jalur 3 : 1-2-3-5-6-10-11-12-19-20-22-23-24

Jalur 4 : 1-5-6-7-9-10-11-12-19-20-22-23-24

Jalur 5 : 1-5-6-7-8-10-11-12-19-20-22-23-24

Jalur 6 : 1-5-6-10-11-12-13-14-17-18-19-20-22-23-24

Jalur 7 : 1-5-6-10-11-12-13-14-15-16-18-19-20-22-23-24

Jalur 8 : 1-5-6-10-11-12-13-14-19-20-22-23-24

Jalur 9 : 1-5-6-10-11-12-19-20-21-22-23-24

Penentuan kasus uji untuk jalur *independent* tersebut dan hasil eksekusinya dijelaskan pada Tabel 5.14.

Tabel 5. 14 *Test Case* Pengujian Operasi Pergerakan Pemain

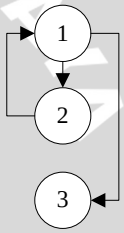
Jalur	Kasus Uji	Hasil yang diharapkan	Hasil yang didapatkan
1	Semua kondisi diberi nilai false.	Player tidak mengalami pergerakan dan posisi x dan y pemain tetap.	Player tidak mengalami pergerakan dan posisi x dan y pemain tetap.
2	Kondisi moveRight true dan Possx diberi nilai lebih besar dari maxRight	Sistem ditandai dengan reachRight true yang berarti pemain berhenti melakukan pergerakan dan digantikan environtment yang melakukan pergerakan.	Sistem ditandai dengan reachRight true yang berarti pemain berhenti melakukan pergerakan dan digantikan environtment yang melakukan pergerakan.
3	Kondisi moveRight true dan Possx diberi nilai lebih kecil dari maxRight	Pemain melakukan pergerakan dengan possx pemain ditambah speed secara berulang – ulang. Dan reachRight ditandai false.	Pemain melakukan pergerakan dengan possx pemain ditambah speed secara berulang – ulang. Dan reachRight ditandai false.
4	Kondisi moveLeft true	Sistem ditandai	Sistem ditandai

	dan Possx diberi nilai lebih besar dari maxLeft	dengan reachLeft true yang berarti pemain berhenti melakukan pergerakan.	dengan reachLeft true yang berarti pemain berhenti melakukan pergerakan.
5	Kondisi moveLeft true dan Possx diberi nilai lebih kecil dari maxLeft	Pemain melakukan pergerakan dengan possx pemain dikurangi speed secara berulang – ulang. Dan reachLeft ditandai false.	Pemain melakukan pergerakan dengan possx pemain dikurangi speed secara berulang – ulang. Dan reachLeft ditandai false.
6	Tidak diberi inputan moveRight ketika Possy pemain lebih besar dari possTanah dan fromJump true	Frame Squence pemain di set idle	Frame Squence pemain di set idle
7	Diberi inputan moveRight ketika possy pemain lebih besar dari possTanah dan fromJump true	Frame Squence pemain di set walk	Frame Squence pemain di set walk
8	fromJump di beri masukan false	Saat posisi turun dari melompat Frame Squence di set idle.	Saat posisi turun dari melompat Frame Squence di set idle.
9	Tidak dilakukan inputan dan posisi pemain onGround	Frame Squence pemain di set idle	Frame Squence pemain di set idle

2. Pengujian Integrasi Operasi Interaksi Musuh

Operasi interaksi musuh merupakan operasi untuk prosedur interaksi musuh, yang diimplementasikan dalam method `cycle()`. Pemodelan operasi interaksi musuh dalam bentuk *flow graph* ditunjukkan pada tabel 5.15.

Tabel 5. 15 Pemodelan *Flow Graph* Operasi Interaksi Musuh

Pseudocode	***Flow Graph***
Methode: cycle **DECLARATION:** feedback IS Boolean value, speedAll, speed IS Int **DESCRIPTION:** 1. FOR (int i=0 TO i<enemies.size()THEN i++)THEN 2. CALL((Enemy)enemies.elementAt(i)).move(feedback,value) 3. END FOR 4. END cycle	Node (N) = 3 Edge (E) = 3 

Pemodelan ke dalam *flow graph* yang telah dilakukan terhadap operasi interaksi musuh menghasilkan jumlah kompleksitas siklomatis *(cyclomatic complexity)* melalui persamaan V(G) = E – N + 2 .

$$V(G) = E - N + 2$$
$$= 3 - 3 + 2$$
$$= 2$$

Berdasarkan dari nilai *cyclomatic complexity* yang telah didapatkan dari perhitungan maka ditentukan satu buah basis set dari jalur *independent*, yaitu :

Jalur 1: 1 – 2 – 1 – 2 – …….

Jalur 2: 1 – 3

Penentuan kasus uji untuk jalur *independent* tersebut dan hasil eksekusinya dijelaskan pada Tabel 5.16.

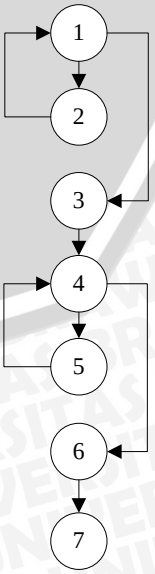
Tabel 5. 16 *Test Case* Pengujian Interaksi Musuh

Jalur	Kasus Uji	Hasil yang diharapkan	Hasil yang didapatkan
1	Iterasi telah sampai batas iterasi	Penambahan enemy telah berhenti.	Penambahan enemy telah berhenti.
2	Jumlah iterasi masih lebih kecil dari jumlah size enemy yang akan ditampilkan	Enemy terus bertambah sesuai posisi yang ditentukan.	Enemy terus bertambah sesuai posisi yang ditentukan.

3. Pengujian Integrasi Operasi Interaksi Item

Operasi interaksi item merupakan operasi untuk prosedur interaksi item *piece puzzle* dan item *health*, yang diimplementasikan dalam method `cycle()`. Pemodelan operasi interasksi item dalam bentuk *flow graph* ditunjukkan pada Tabel 5.17.

Tabel 5. 17 Pemodelan *Flow Graph* Operasi Interaksi Item

Pseudocode	***Flow Graph***
Methode: cycle **DECLARATION:** feedback IS Boolean value, speedAll, speed IS Int **DESCRIPTION:** 1. FOR (int i=0 TO i< batikItems.size()THEN i++)THEN 2. CALL ((Enemy) batikItems.elementAt(i)).muncul() 3. END FOR 4. FOR (int i=0 TO i< healthItems.size()THEN i++)THEN 5. CALL((Enemy) healthItems.elementAt(i)).muncul() 6. END FOR 5. END cycle	Node (N) = 7 Edge (E) = 8 

Pemodelan ke dalam *flow graph* yang telah dilakukan terhadap operasi jump ( ) menghasilkan jumlah kompleksitas siklomatis *(cyclomatic complexity)* melalui persamaan V(G) = E – N + 2 .

V(G) = E – N + 2

= 8 – 7 + 2

= 3

Berdasarkan dari nilai *cyclomatic complexity* yang telah didapatkan dari perhitungan maka ditentukan satu buah basis set dari jalur *independent*, yaitu :

Jalur 1: 1 – 3 – 4 – 6 – 7

Jalur 2: 1 – 2 – 1 – …….

Jalur 3 : 1 – 3 – 4 – 5 – 4 – …….

Penentuan kasus uji untuk jalur *independent* tersebut dan hasil eksekusinya dijelaskan pada Tabel 5.18.

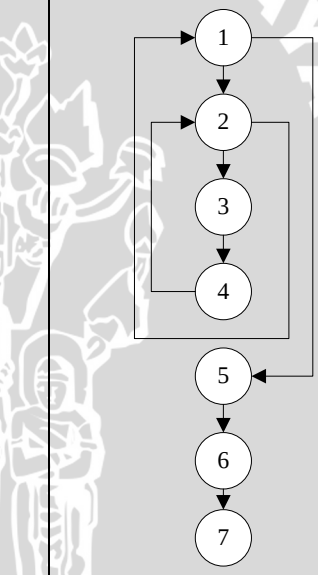
Tabel 5. 18 *Test Case* Pengujian Operasi Interaksi Item

Jalur	Kasus Uji	Hasil yang diharapkan	Hasil yang didapatkan
1	Iterasi telah sampai batas iterasi	Penambahan item telah berhenti.	Penambahan item telah berhenti.
2	Jumlah iterasi lebih kecil dari jumlah batikItem	Potongan item *puzzle* terus bertambah sesuai posisi yang ditentukan.	Potongan item *puzzle* terus bertambah sesuai posisi yang ditentukan.
3	Jumlah iterasi lebih besar dari jumlah healthItem	Bonus item nyawa terus bertambah sesuai posisi yang ditentukan.	Bonus item nyawa terus bertambah sesuai posisi yang ditentukan.

4. Pengujian Integrasi Operasi Pengengecekan *Puzzle*

Operasi pengecekan *puzzle* merupakan operasi untuk prosedur modul kontrol pengecekan kondisi menang pada permainan *puzzle*, yang diimplementasikan dalam method `winCheck()`. Pemodelan operasi dalam bentuk *flow graph* ditunjukkan pada Tabel 5.19.

Tabel 5. 19 Pemodelan *Flow Graph* Operasi Pengecekan *Puzzle*

Pseudocode	*Flow Graph*
Methode: winCheck **DECLARATION:** grid IS Int **DESCRIPTION:** 1. FOR (int i=0 TO i < grid THEN i++) 2. FOR (int j=0 TO j < grid THEN j++) 3. IF (!tiles[i][j].tileMatch()) 4. return FALSE 5. END FOR 6. END FOR 7. return TRUE 8. END winCheck	Node (N) = 7 Edge (E) = 8 

Pemodelan ke dalam *flow graph* yang telah dilakukan terhadap operasi jump() menghasilkan jumlah kompleksitas siklomatis *(cyclomatic complexity)* melalui persamaan V(G) = E – N + 2 .

V(G) = E – N + 2

= 8 – 7 + 2

= 3

Berdasarkan dari nilai *cyclomatic complexity* yang telah didapatkan dari perhitungan maka ditentukan satu buah basis set dari jalur *independent*, yaitu :

Jalur 1: 1 – 5 – 6 – 7

Jalur 2: 1 – 2 – 1 – ……

Jalur 3: 1 – 2 – 3 – 4 – 2 – 1 – ….

Penentuan kasus uji untuk jalur *independent* tersebut dan hasil eksekusinya dijelaskan pada Tabel 5.20.

Tabel 5. 20 *Test Case* Pengujian operasi Pengecekan *Puzzle*

Jalur	Kasus Uji	Hasil yang diharapkan	Hasil yang didapatkan
1	Iterasi telah sampai batas iterasi.	Mengembalikan nilai true yang berarti pengecekan telah selesai *puzzle* telah tersusun dengan benar.	Mengembalikan nilai true yang berarti pengecekan telah selesai *puzzle* telah tersusun dengan benar.
2	Pengecekan kondisi j telah sampai batas iterasi.	Melakukan pengecekan dari i lagi.	Melakukan pengecekan dari i lagi.
3	Pengecekan kondisi masih memenuhi batas iterasi.	Mengulang iterasi sampai batas iterasi untuk melakukan pengecekan kondisi menang.	Mengulang iterasi sampai batas iterasi untuk melakukan pengecekan kondisi menang.

5.1.3 Pengujian Validasi

Pengujian validasi digunakan untuk mengetahui apakah sistem yang dibangun sudah benar sesuai dengan yang dibutuhkan. *Item - item* yang telah dirumuskan dalam daftar kebutuhan dan merupakan hasil analisis kebutuhan akan menjadi acuan untuk melakukan pengujian validasi. Pengujian validasi menggunakan metode pengujian

Black Box, karena tidak diperlukan konsentrasi terhadap alur jalannya *Prosedur* program dan lebih ditekankan untuk menemukan konformitas antara kinerja sistem dengan daftar kebutuhan.

5.1.3.1 Kasus Uji Validasi

Keseluruhan kasus uji validasi dijelaskan pada tabel 5.21.

Tabel 5. 21 Kasus Uji Validasi

No.	Kasus Uji	Objek Uji	Prosedur dan Input	Kondisi yang diharapkan	Kondisi Gagal
1.	Melihat *Splash Screen*	F01	Pemain menekan ikon permainan untuk memulai eksekusi *game*.	Aplikasi dapat menampilkan *splash screen* 1,2,3 secara berurutan.	Aplikasi tidak dapat menampilkan *splash screen*. Atau dapat menampilkan tetapi tidak sesuai urutan.
2.	Melihat cerita	F02	Pemain melakukan sentuh geser kekiri dan kekanan pada screen.	Aplikasi secara otomatis menampilkan cerita halaman pertama sesaat setelah *Splash Screen* selesai dieksekusi. Saat berada di halaman pertama jika di geser ke kiri *screen* akan berganti ke	Aplikasi tidak dapat menampilkan halaman cerita. Ketika dilakukan proses geser, screen tidak berpindah.

				cerita halaman kedua. Saat di geser ke kiri lagi *screen* akan berganti ke halaman main menu	
3.	Melihat menu utama	F03	Pemain melakukan proses geser kiri pada halaman kedua screen cerita.	Aplikasi dapat menampilkan halaman menu utama.	Aplikasi tidak dapat menampilkan halaman menu utama.
4.	Memulai permainan	F04	Pemain memilih menu play dengan menekan tombol dengan icon play.	Aplikasi dapat menampilkan halaman memilih level permainan.	Aplikasi tidak dapat menampilkan halaman memilih level permainan.
5.	Memilih level permainan.	F05	Pemain memilih level 1 dengan menekan gambar tombol pekalongan.	Aplikasi menampilkan *gameplay* level 1.	Aplikasi tidak menampilkan *gameplay* level 1.
			Pemain memilih level 2 dengan menekan gambar tombol tasikmalaya.	Aplikasi menampilkan *gameplay* level 2.	Aplikasi tidak menampilkan *gameplay* level 2.

			Pemain memilih level 3 dengan menekan gambar tombol garut.	Aplikasi menampilkan *gameplay* level 3.	Aplikasi tidak menampilkan *gameplay* level 3.
			Pemain memilih level 4 dengan menekan gambar tombol solo.	Aplikasi menampilkan *gameplay* level 4.	Aplikasi tidak menampilkan *gameplay* level 4.
			Pemain memilih level 5 dengan menekan gambar tombol yogyakarta.	Aplikasi menampilkan *gameplay* level 5.	Aplikasi tidak menampilkan *gameplay* level 5.
			Pemain memilih level 6 dengan menekan gambar tombol cirebon.	Aplikasi menampilkan *gameplay* level 6.	Aplikasi tidak menampilkan *gameplay* level 6.
6.	Melakukan proses pause.	F06	Pemain melakukan perintah pause pada *in game*.	Aplikasi menghentikan sementara permainan yang berlangsung lalu menampilkan	Aplikasi tidak menghentikan sementara permainan yang berlangsung sehingga tidak menampilkan

				menu pause yang memberikan pilihan untuk kembali ke permainan, reset atau keluar dari game.	menu pause yang memberikan pilihan untuk kembali ke permainan atau keluar dari game.
7	Memperoleh konfirmasi ke permainan *puzzle*	F07	Pemain berhasil mengumpulkan semua *piece puzzle*.	Aplikasi menampilkan dialog konfirmasi menuju permainan *puzzle*.	Aplikasi tidak dapat menampilkan dialog konfirmasi menuju permainan *puzzle*.
8	Masuk Permainan *Puzzle*	F08	Pemain menekan tombol konfirmasi ke permainan *puzzle* setelah berhasil mengumpulkan potongan *puzzle* di permainan utama.	Aplikasi menampilkan permainan *puzzle* sesuai level pada permaian utama.	Aplikasi tidak dapat menampilkan permainan *puzzle* sesuai level pada permainan utama.
9	Memperoleh dialog berhasil	F09	Pemain berhasil menyusun	Aplikasi menampilkan dialog	Aplikasi tidak dapat menampilkan

	menyelesaikan permainan dan konfirmasi ke halaman galeri.		*puzzle* menjadi gambar yang utuh.	konfirmasi menuju halaman galeri.	dialog komfirmasi menuju halaman galeri.
10	Melihat galeri.	F10	Pemain memilih menu galeri dengan menekan tombol icon galeri.	Aplikasi dapat menampilkan halaman galeri.	Aplikasi tidak dapat menampilkan halaman galeri.
11	Melihat deskripsi batik.	F11	Pemain menekan tombol icon deskripsi yang dinginkan setelah halaman galeri terbuka.	Aplikasi dapat menampilkan deskripsi batik sesuai tombol yang dipilih pemain.	Aplikasi tidak dapat menampilkan deskripsi batik sesuai tombol yang dipilih pemain.
12	Mengatur suara	F12	Pemain memilih menu pengaturan suara dengan menekan icon suara.	Apliakasi dapat melakukan pengaturan suara on atau off.	Aplikasi tidak dapat melakukan pengaturan suara.
13	Melihat cara bermain.	F13	Pemain memilih menu melihat cara bermain dengan	Aplikasi menampilkan halaman cara bermain.	Aplikasi tidak dapat menampilkan halaman cara bermain.

			menekan icon tombol help.		
14	Melihat info pembuat *game*.	F14	Pemain memilih menu melihat info dengan menekan icon tombol credits.	Aplikasi menampilkan halaman info pembuat *game*.	Aplikasi tidak dapat menampilkan halaman info pembuat *game*.
15	Keluar permainan.	F08	Pemain menekan icon tombol keluar.	Keluar dari aplikasi.	Tidak keluar dari aplikasi.

5.1.3.2 Hasil Pengujian Validasi

Hasil pengujian validasi dijelaskan pada Tabel 5.22.

Tabel 5. 22 Hasil Uji Validasi

No.	Kasus Uji	Objek Uji	Prosedur dan Input	Kondisi yang diharapkan	Hasil
1.	Melihat *Splash Screen*	F01	Pemain menekan ikon permainan untuk memulai eksekusi *game*.	Aplikasi dapat menampilkan *splash screen* 1,2,3 secara berurutan.	Valid
2.	Melihat cerita	F02	Pemain melakukan sentuh geser kekiri dan kekanan pada screen.	Aplikasi secara otomatis menampilkan cerita halaman pertama sesaat setelah *Splash Screen* selesai dieksekusi. Saat berada di halaman pertama	Valid

				jika di geser ke kiri *screen* akan berganti ke cerita halaman kedua. Saat di geser ke kiri lagi *screen* akan berganti ke halaman main menu	
3.	Melihat menu utama	F03	Pemain melakukan proses geser kiri pada halaman kedua screen cerita.	Aplikasi dapat menampilkan halaman menu utama.	Valid
4.	Memulai permainan	F04	Pemain memilih menu play dengan menekan tombol dengan icon play.	Aplikasi dapat menampilkan halaman memilih level permainan.	Valid
5.	Memilih level permainan.	F05	Pemain memilih level 1 dengan menekan gambar tombol pekalongan.	Aplikasi menampilkan *gameplay* level 1.	Valid
			Pemain memilih level 2 dengan menekan gambar tombol tasikmalaya.	Aplikasi menampilkan *gameplay* level 2.	Valid
			Pemain memilih level 3 dengan menekan gambar tombol garut.	Aplikasi menampilkan *gameplay* level 3.	Valid

			Pemain memilih level 4 dengan menekan gambar tombol solo.	Aplikasi menampilkan *gameplay* level 4.	Valid
			Pemain memilih level 5 dengan menekan gambar tombol yogyakarta.	Aplikasi menampilkan *gameplay* level 5.	Valid
			Pemain memilih level 6 dengan menekan gambar tombol cirebon.	Aplikasi menampilkan *gameplay* level 6.	Valid
6.	Melakukan proses pause.	F06	Pemain melakukan perintah pause pada *in game*.	Aplikasi menghentikan sementara permainan yang berlangsung lalu menampilkan menu pause yang memberikan pilihan untuk kembali ke permainan, reset atau keluar dari game.	Valid
7	Memperoleh konfirmasi ke permainan *puzzle*	F07	Pemain berhasil mengumpulkan semua *piece puzzle*.	Aplikasi menampilkan dialog konfirmasi menuju permainan *puzzle*.	Valid
8	Masuk Permainan *Puzzle*	F08	Pemain menekan tombol konfirmasi ke permainan *puzzle* setelah	Aplikasi menampilkan permainan *puzzle* sesuai level pada permaian utama.	Valid

			berhasil mengumpulkan potongan *puzzle* di permainan utama.		
9	Memperoleh dialog berhasil menyelesaikan permainan dan konfirmasi ke halaman galeri.	F09	Pemain berhasil menyusun *puzzle* menjadi gambar yang utuh.	Aplikasi menampilkan dialog konfirmasi menuju halaman galeri.	Valid
10	Melihat galeri.	F10	Pemain memilih menu galeri dengan menekan tombol icon galeri.	Aplikasi dapat menampilkan halaman galeri.	Valid
11	Melihat deskripsi batik.	F11	Pemain menekan tombol icon deskripsi yang dinginkan setelah halaman galeri terbuka.	Aplikasi dapat menampilkan deskripsi batik sesuai tombol yang dipilih pemain.	Valid
12	Mengatur suara	F12	Pemain memilih menu pengaturan suara dengan menekan icon suara.	Apliakasi dapat melakukan pengaturan suara on atau off.	Valid
13	Melihat cara bermain.	F13	Pemain memilih menu melihat cara bermain dengan	Aplikasi menampilkan halaman cara bermain.	Valid

			menekan icon tombol help.		
14	Melihat info pembuat *game*.	F14	Pemain memilih menu melihat info dengan menekan icon tombol credits.	Aplikasi menampilkan halaman info pembuat *game*.	Valid
15	Keluar permainan.	F08	Pemain menekan icon tombol keluar.	Keluar dari aplikasi.	Valid

5.1.4 Pengujian Performa

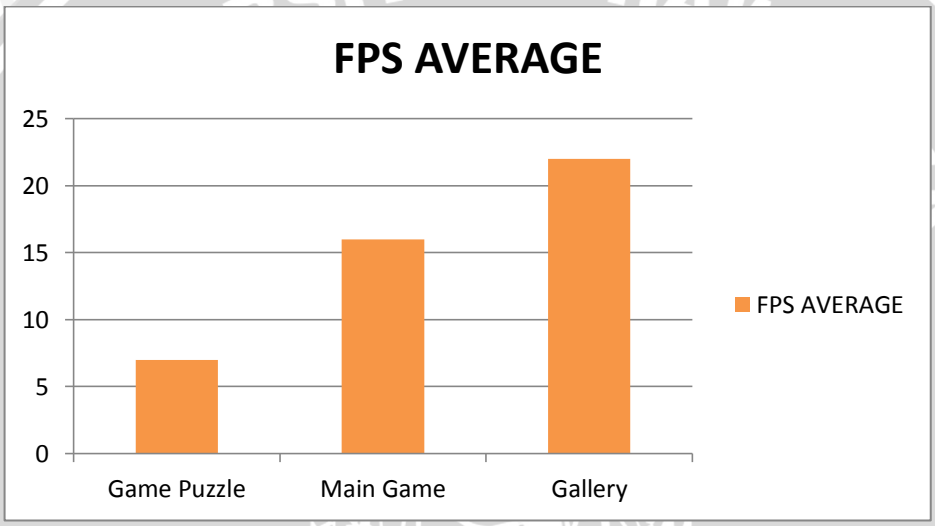
Pengujian performa *game* Pengenalan Budaya Batik dilakukan dengan melihat FPS (*Frame Per Second*) yang dihasilkan oleh *game*. Pengujian FPS dilakukan pada perangkat *mobile* Nokia Asha. Pengujian kinerja dilakukan dengan melihat FPS yang dihasilkan pada setiap halaman. Adapun halaman yang digunakan dalam pengujian ini adalah *main game*, *game puzzle*, *gallery*. Hasil pengujian kinerja *game* ditunjukkan pada Tabel 5.23. Tabel 5.24 menunjukan pengaruh FPS terhadap kinerja *game*. Gambar 6.14 menunjukan grafik FPS dari tiap halaman *game* yang dihasilkan dari pengujian.

Tabel 5. 23 Hasil pengujian kinerja dengan melihat FPS

Spesifikasi Sistem	**FPS**		
	Main Game	**Game *Puzzle***	**Gallery**
Nokia Asha 311, RAM 256 MB, Maximum Heap Size 4 MB, Maximum JAR Size 2 MB.	16-17	6-9	21-23

Tabel 5. 24 Pengaruh FPS Terhadap kinerja *Game*

FPS	Kinerja *Game*
FPS < 10	Game berjalan lebih lambat
FPS > 10 dan FPS < 20	Game berjalan normal
FPS > 20	Game berjalan cepat



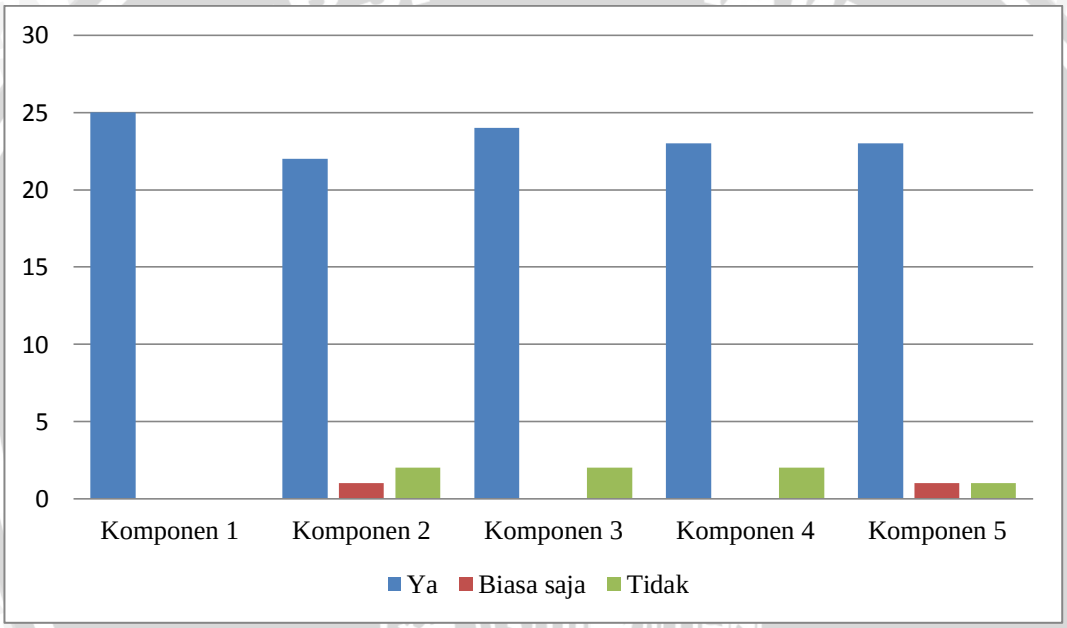
Gambar 5. 1 Grafik FPS AVERAGE *Game*

5.1.5 Pengujian Terhadap Pengguna

Pengujian kepada pengguna dilakukan untuk memastikan bahwa pembuatan *game* telah sesuai dengan tujuan yang ingin dicapai yaitu *game* yang menyenangkan untuk memperkenalkan budaya batik Indonesia. Pengujian dilakukan dengan pembuatan kuesioner. Parameter kuesioner pengujian hasil ditunjukkan pada Tabel 5.25. Grafik hasil pengujian ditunjukkan pada Gambar 5.2. Umpan balik pengujian ditunjukkan pada Tabel 5.26. Umpan balik dari pengguna dicatat dalam kuisioner sebagai saran yang ditulis oleh pemain.

Tabel 5. 25 Komponen Kuesioner

No. Komponen	Isi Komponen Kuesioner
1.	Apakah *game* menyenangkan?
2.	Apakah *game* menambah pengetahuan tentang batik?
3.	Apakah tampilan *game* menarik?
4.	Apakah *game* mudah dimainkan?
5.	Apakah *game* cocok untuk semua usia?



Gambar 5. 2 Grafik hasil pengujian kuesioner

Tabel 5. 26 Umpan balik saran pengguna

No.	Saran
1.	Dalam pengembangan selanjutnya dapat dilakukan untuk pengenalan batik dari semua daerah di Indonesia.
2.	Dalam pengembangan selanjutnya dapat dilakukan penambahan fitur-fitur permainan.

5.2 Analisis

Proses analisis bertujuan untuk mendapatkan kesimpulan dari hasil pengujian *game* Pengenalan Budaya Batik Indonesia yang telah dilakukan. Proses analisis mengacu pada dasar teori sesuai dengan hasil pengujian yang didapatkan. Analisis dilakukan terhadap hasil pengujian di setiap tahap pengujian. Proses analisis yang dilakukan meliputi analisis hasil pengujian unit, analisis hasil pengujian integrasi, analisis hasil pengujian validasi, dan analisis hasil pengujian performa.

5.2.1 Analisis Hasil Pengujian Unit

Proses analisis terhadap hasil pengujian unit dilakukan dengan melihat kesesuaian fungsi dari implementasi unit modul yang diuji dengan hasil perancangan *game* yang telah dirancang sebelumnya. Dari seluruh pengujian *unit* yang dilakukan, jumlah jalur pada logika setiap *method* telah sesuai dengan perhitungan *cyclomatic complexity*. Setiap kasus uji yang dibuat berdasarkan jumlah jalur tersebut telah diuji dan memberikan hasil sesuai dengan hasil yang diharapkan. Berdasarkan hal tersebut, maka dapat diambil kesimpulan bahwa unit modul dari program sudah memenuhi kebutuhan fungsional yang telah dirancang pada tahap perancangan.

5.2.2 Analisis Hasil Pengujian Integrasi

Proses analisis terhadap hasil pengujian integrasi dilakukan dengan melihat kesesuaian beberapa unit modul yang menyusun satu blok fungsi dalam *game*. Dari seluruh pengujian integrasi yang dilakukan, jumlah jalur pada logika setiap prosedur telah sesuai dengan perhitungan *cyclomatic complexity*. Setiap kasus uji yang dibuat berdasarkan jumlah jalur tersebut telah diuji dan memberikan hasil sesuai dengan hasil yang diharapkan. Berdasarkan hal tersebut, maka dapat diambil kesimpulan bahwa unit modul dari program sudah memenuhi kebutuhan fungsional yang telah dirancang pada tahap perancangan.

5.2.3 Analisis Hasil Pengujian Validasi

Berdasarkan kesesuaian antara hasil uji terhadap implementasi dan fungsionalitas *game* Pengenalan Budaya Batik Indonesia dengan hasil yang diharapkan dalam daftar kebutuhan *game*, melalui pengujian menggunakan Emulator Nokia SDK 2.0 didapatkan keseluruhan fungsionalitas *game* telah memenuhi kebutuhan yang telah dijabarkan dalam daftar kebutuhan. Namun ketika game diinstal pada *device*, tidak semua fungsional game berjalan dengan baik, hal ini dikarenakan perbedaan spesifikasi *device*. Semakin tinggi spesifikasi *device*, semakin sedikit *bug* yang ditemukan. Semakin rendah spesifikasi *device* semakin banyak *bug* yang ditemukan. *Device* yang paling kompatible dengan *game* ini adalah nokia Asha dengan spesifikasi RAM 256 MB, maximum Heap Size 4 MB dan maximum JAR Size 2 MB.

5.2.4 Analisis Hasil Pengujian Performa

Hasil analisa yang didapatkan dari pengujian performa yang telah dilakukan, rata-rata FPS normal yang didapat adalah 10 – 20. FPS rata-rata mobile game J2ME berbeda dengan FPS rata-rata game PC. Dikarenakan perbedaan spesifikasi memori dan prosesor. Berdasarkan pengujian FPS tercepat berada pada halaman galeri yaitu 21 – 23 FPS, dan FPS terlambat berada pada halaman Permainan *Puzzle* yaitu 6 – 9 FPS. Hal ini dipengaruhi seberapa besar komponen yang harus di proses oleh *game*. Sehingga dari hasil pengujian di dapatkan bahwa semakin kecil komponen yang diproses semakin besar pula hasil FPS yang di dapat dan semakin halus tampilan game.

5.2.5 Analisis Hasil Pengujian Terhadap Pengguna

Hasil yang didapat melalui pengujian kuesioner terhadap 25 responden didapatkan data sebagai berikut, dari pengujian komponen satu yang berisi apakah game yang dibangun menyenangkan didapatkan hasil bahwa 100% responden menyatakan bahwa *game* menyenangkan. Dari pengujian komponen dua yang berisi apakah *game* yang dibangun menambah pengetahuan tentang budaya batik didapatkan 88% responden menyatakan bahwa *game* menambah pengetahuan pengguna tentang budaya batik, sedangkan 4% responden menyatakan bahwa

game tidak penambah pengetahuan tentang budaya batik. Analisis terhadap responden yang menyatakan game yang dibangun tidak menambah pengetahuan tentang budaya batik dikarenakan ada sebagian responden yang tidak memainkan *game* sampai selesai, sehingga tujuan *game* tidak tercapai. Dari pengujian komponen ketiga yang berisi apakah tampilan game yang dibangun sudah menarik didapatkan 96% responden menyatakan bahwa tampilan game sudah menarik. Dari pengujian komponen keempat yang berisi apakah game mudah dimainkan user didapatkan 92% responden menyatakan bahwa game mudah dimainkan. Dari pengujian komponen kelima yang berisi apakah game yang dibangun cocok untuk semua usia didapatkan 92% responden menyatakan bahwa game yang dibangun cocok untuk semua usia. Dari hasil keseluruhan pengujian terhadap pengguna didapatkan kesimpulan bahwa perancangan dan pembangunan game telah sesuai dengan tujuan yang diharapkan yaitu mengembangkan game yang menyenangkan untuk memperkenalkan budaya batik Indonesia.

BAB VI
PENUTUP

6.1 Kesimpulan

Berdasarkan hasil perancangan, implementasi dan pengujian yang dilakukan, maka diambil kesimpulan sebagai berikut :

1. Batik Mania adalah sebuah game *platformer* dan *puzzle* dalam satu aplikasi dengan mengusung tema edukasi pengenalan budaya batik Indonesia.
2. Batik Mania telah berhasil diimplentasikan sebagai game pengenalan budaya batik, dimana unsur pengenalan terdapat pada art dan deskripsi batik pada bonus material.
3. Berdasarkan hasil pengujian unit dan pengujian integrasi dengan menggunakan metode *whitebox testing*, didapatkan jumlah jalur pada logika setiap *method* dan prosedur telah sesuai dengan perhitungan *cyclomatic complexity* dan setiap jalur *independent* dengan hasil pengujian sesuai dengan hasil yang diharapkan.
4. Berdasarkan hasil pengujian validasi menggunakan metode *blackbox testing*, didapatkan keseluruhan fungsional aplikasi permainan dapat berjalan sesuai daftar kebutuhan yang telah dibuat dan dimodelkan dalam *use case*.
5. Berdasarkan hasil pengujian performa dengan melihat FPS rata-rata yang dihasilkan *game* adalah antara 10 – 20 FPS dan telah sesuai dengan standart FPS *game* J2ME.

6.2 Saran

Saran untuk pengembangan game pengenalan budaya batik Indonesia lebih lanjut antara lain :

1. Dalam mengembangkan *game* berbasis J2ME harus memperhatikan pemakaian memori, untuk memperkecil penggunaan memori dapat dilakukan dengan memperkecil ukuran aset *game* dan melakukan proses *destroy* setiap program yang telah dieksekusi tidak lagi digunakan.
2. Dalam pengembangan game pengenalan budaya batik selanjutnya dapat dilakukan dengan platform lain.

3. Dalam pengembangan selanjutnya dapat dilakukan untuk pengenalan batik dari semua daerah di Indonesia.
4. Dalam pengembangan selanjutnya dapat dilakukan penambahan fitur permainan, seperti penambahan level permainan, penambahan tingkat kesulitan *gameplay*, penambahan bonus material tentang batik.
5. Dalam pengembangan selanjutnya dapat dibangun *game* pengenalan budaya batik dengan *genre game* yang berbeda.

PUSTAKA

[ALH-03] Alhir, Sinan Si. 2003, "Learning UML", O'Reilly Media, Inc.

[ANO-13] Anonim. 2013. "Business Opportunities". http://www.developer.nokia.com/Develop/Series_40/Business_opportunity/ [4 Maret 2013].

[AN1-13] Anonim. 2013. "Stat Counter Global Stat". http://gs.statcounter.com/ [7 Maret 2013].

[DWI-12] Dwiperdana, Aditya, Ardika Prasetyo, dan Henky Jaya Dinata. 2012. "Membuat *Game* J2ME untuk Pemula", Agate, Bandung.

[FOX-02] Fox, David. 2002. "Micro Java™ Game Development", Addison Wesley, US.

[KAR-10] Karmila, Mila. 2010. "Ragam Kain Tradisional Nusantara (Makna, Simbol, dan Fungsi)", Bee Media Indonesia, Jakarta.

[MUN-05] Munawar. 2005. "Pemodelan Visual dengan UML", Graha Ilmu, Yogyakarta.

[PRE-10] Pressman, Roger S. 2010. "Software Engineering : A Practitioner's Approach, Seventh Edition". McGraw Hill.

[PUR-10] Purnama, Rangsang. 2012. "Mari Mengenal J2ME Java 2 Micro Edition", Prestasi Pustaka, Jakarta.

[ROG-10] Rogers, Scott. 2010. "Level Up! The Guide to Great Video Game Design", Willey, US.

[WAH-12] Wahyu, Ami. 2012. "Chic in Batik", Erlangga, Jakarta.