APLIKASI PERMAINAN *FIND THE PAIR* DENGAN METODE PENCAIRAN *ITERATIVE DEEPENING SEARCH*

Setiyo, Mochammad Rizki Romdoni
Mahasiswa STT Indonesia Tanjungpinan,
Ketua Program Studi Sistem Informasi
STT Indonesia Tanjungpinang
Setiyo@gmail.com, rizki@sttindonesia.ac.id

Abstract

Permainan merupakan suatu sarana hiburan yang diminati dan dimainkan oleh banyak orang, baik dari kalangan anak-anak, remaja maupun orang dewasa. Permainan terdiri dari permainan tradisional dan permainan modern. Banyak diantara penggemar permainan lebih tertarik dengan permainan modern, hal ini dikarenakan permainan modern bisa dimainkan dimana saja secara praktis dan mudah. Banyak sekali permainan modern yang telah berkembang salah satunya adalah permainan Find the pair.

Permainan Find the pair adalah permainan kartu di mana semua kartu diletakkan menghadap ke bawah pada permukaan dan dua kartu dibalikkan menghadap ke atas pada setiap giliran untuk mencari pasangan dari kartu sebelumnya. Tujuan dari permainan Find the pair adalah melatih kosentrasi dan daya ingat bagi yang memainkannya, tampilan permainan ini dibuat semenarik mungkin dengan adanya gambar yang berhubungan kebudayaan dan kesenian indonesia.

Find the pair ini di buat dengan beberapa metode agar bisa berjalan dengan baik. Untuk sistem pencarian menggunakan metode iterative deepening search. Untuk pengacakan menggunakan pengacakan sistematis, karena sistemnya lebih praktis dan hemat waktu. Dalam permainan ini akan dilakukan penilaian agar permainan ini menarik dan banyak dimainkan. Software yang digunakan dalam pembuatan Skripsi ini adalah Lazarus dan metodologi yang digunakan untuk perkembangan aplikasi ini adalah menggunakan paradigma Sekuensial Linier. Permainan Find the pair ini bisa dimainkan di komputer dan laptop.

Keywords : Permainan Find the Pair, Lazarus, Sekuensial Linier

I. PENDAHULUAN

A. Latar Belakang Penelitian

Permainan merupakan suatu sarana hiburan yang diminati dan dimainkan oleh banyak orang baik dari kalangan anak-anak, remaja maupun orang dewasa. Permainan ini terdiri dari permainan tradisional dan permainan *modern*. Karena mereka dapat bermain permainan dimana saja secara praktis dan mudah. Kesederhanaan dalam bermain permainan namun tidak membosankan, serta dapat membuat pemain ingin kembali memainkan permainan tersebut.

Oleh karena itu berbagai pendekatan terus dikembangkan untuk membuat sebuah permainan yang dapat dimainkan dalam berbagai perangkat.

Menurut Jean Piaget pada teorinya *Cognitive-Developmental* yang dikutip oleh McLeod (2009), mengungkapkan bahwa bermain dapat mengaktifkan otak, dengan mengintegrasikan fungsi otak kanan dan otak kiri secara seimbang dan membentuk struktur saraf, serta mengembangkan pilar saraf pemahaman yang berguna untuk masa mendatang. Disamping itu, otak yang aktif adalah kondisi dimana seseorang akan sangat baik untuk menerima pelajaran.

Berdasarkan kajian tersebut maka bermain merupakan kegiatan yang sangat penting bagi manusia karena melalui bermain seseorang dapat mengembangkan aspek-aspek seperti fisik, motorik dan kognitif. Kegiatan bermain yang dapat dilakukan untuk menunjang perkembangan ini antara lain seperti menggambar dan pengenalan objek-objek baru dengan gambar yang dibuat unik agar mudah diingat.

Berdasarkan latar belakang tersebut, maka timbul keinginan untuk mengembangkan inovasi pada permainan dengan media komputer. *Game* yang akan dikembangkan tersebut merupakan sebuah permainan yang dikenal dengan nama "*Find the Pair*". Permainan ini dapat melatih kemampuan konsentrasi dan daya ingat. permainan ini akan berisikan pengenalan gambar yang menarik, misalnya yang berhubungan dengan artis, *anime*, ataupun gambar makanan tertentu.

B. Rumusan Masalah

Berdasarkan latar belakang dan batasan masalah diatas, maka rumusan masalah adalah sebagai berikut :

1. Bagaimana cara membuat aplikasi permainan dengan menampilkan unsur yang berhubungan dengan Indonesia ?
2. Bagaimana menerapkan sistem pencarian berdasarkan metode *iterative deepening search* ?
3. Bagaimana menerapkan aplikasi permainan dengan metode acak secara sistematik ?

II. KAJIAN PUSTAKA

A. Kajian Pustaka

1. Perangkat Lunak (*Software*)

Perangkat lunak (*software*) adalah istilah khusus untuk data yang diformat, dan disimpan secara digital, termasuk program komputer, dokumentasinya, dan berbagai informasi yang bisa dibaca, dan ditulis oleh komputer. Dengan kata lain, bagian sistem komputer yang tidak berwujud. Istilah ini menonjolkan perbedaan dengan perangkat keras komputer.

2. Karakteristik Perangkat Lunak

Perangkat lunak mempunyai karakteristik sebagai berikut:

1. Sistem elemen peran gkat lunak bersifat logika, berbeda dengan perangkat keras yang mempunyai sistem elemen yang bersifat fisik. Perangkat lunak dikemban gkan atau di rekayasa. Kualitas perangkat lunak yang baik dapat dicapai melalui rancangan yang baik pula.
2. Biaya pembuatan perangkat lunak difokuskan pada keahlian, ini berarti proyek-proyek perangkat lunak tidak dapat dikelola seperti proyek-proyek lainnya.
3. Perangkat lunak tidak mudah rusak, berbeda dengan perangkat keras yang dapat rusak karena debu, getaran, suhu yang terlalu tinggi atau rendah, dan banyak lagi masalah lainnya.

B. Permainan *Find The Pair*

Permainan *find the pair* juga dikenal sebagai permainan konsentrasi, adalah permainan kartu di mana semua kartu diletakkan menghadap ke bawah pada permukaan seperti meja atau lantai, kemudian dua kartu dibalikkan ke atas pada setiap giliran. Objek dari permainan ini adalah untuk menyamakan pasangan kartu yang cocok.

Hal ini umum bagi banyak pemain untuk berpikir mereka tahu di mana pasangan berada dan untuk menyerahkan itu pada keyakinan mereka, kemudian akan bingung menemukan pasangannya. Strategi yang lebih baik adalah untuk menyerahkan kartu yang belum diketahui pertama, sehingga jika salah, yang tahu untuk tidak repot-repot mengubah kartu menjadi lebih pasti.

Strategi yang ideal dapat dikembangkan jika kita menganggap bahwa pemain memiliki memori yang sempurna. Untuk variasi terbalik menghadap ke bawah, strategi ini cukup sederhana. Sebelum pergantian dalam permainan, ada kartu masih dalam bermain, dan n kartu masih dalam bermain, tapi nilai yang diketahui. Pemain saat ini harus membalikkan ke atas kartu yang tidak diketahui. Jika kartu ini cocok dengan salah satu kartu yang dikenal,

pertandingan yang berikutnya dipilih. Kurang jelas, jika kartu tidak dapat ditemukan pada kartu diketahui, salah satu kartu n dikenal tetap harus dipilih untuk meminimalkan informasi yang diberikan.

Jika kartu tidak cocok mereka kemudian dibalik kembali ke keadaan semula. Berikutnya memilih kartu pertama mereka dan kembali mencari psangan dari kartu yang dipilih. Jika itu adalah pertandingan untuk salah satu kartu pemain sebelumnya diserahkan kemudian mereka mencoba untuk mengingat di mana kartu yang cocok adalah dan mengubahnya. Jika mereka berhasil membuat pertandingan mereka menempatkan kartu di tumpukan mereka dan memilih kartu lain. Permainan terus berlangsung dalam mode ini sampai semua kartu yang dimainkan berhasil ditumpuk terbuka semua.

Jika kartu tidak diketahui sisanya dipilih secara acak, ada 1 / (t-1-n) kesempatan untuk mendapatkan pertandingan, tetapi juga / (t-1-n) kesempatan memberikan lawan dengan informasi yang dibutuhkan untuk membuat pertandingan. Ada beberapa pengecualian untuk aturan ini yang berlaku pada kasus pinggiran, di mana n = 0 atau 1 atau menjelang akhir pertandingan.

Konsentrasi tidak terbatas pada bermain kartu. Banyak versi dari permainan yang dirancang untuk anak-anak mungkin memiliki tema yang berbeda. Dalam beberapa versi komputer, kartu mungkin secara acak bergerak untuk meningkatkan kesulitan.

C. *Iterative Deepening Search*

Dalam ilmu komputer, *iterative deepening search* (IDS) atau lebih khusus dikenal sebagai *iterative deepening depth-first search* (IDDFS) adalah ruang keadaan / grafik strategi pencarian di mana versi kedalaman terbatas pencarian pertama kedalaman dijalankan berulang kali dengan meningkatnya kedalaman batas sampai tujuan ditemukan. IDDFS setara dengan pencarian luas-pertama, tetapi menggunakan lebih sedikit memori, pada setiap iterasi, itu mencari node di pohon pencarian di urutan yang sama seperti *depth-first search*, tetapi urutan kumulatif yang node pertama yang dikunjungi menjadi lebih efektif dibanding *breadth first search*.

IDDFS menggabungkan ruang-efisiensi mendalam pertama pencarian dan kelengkapan *breadth first search* (ketika faktor percabangan terbatas). Ini menjadi lebih optimal ketika waktu pencarian merupakan fungsi yang tidak menurun dari kedalaman node. Kompleksitas waktu dari IDDFS adalah O(b^d) dan yang kompleksitas ruang adalah O(d) , di mana b adalah faktor bercabang dan d adalah kedalaman tujuan yang lebih mendalam.

Pada *iterative deepening search* merupakan kunjungan sampai beberapa kali, mungkin tampak boros waktu, tapi ternyata menjadi tidak begitu, karena di pohon sebagian besar node berada di tingkat bawah, sehingga tidak peduli banyak jika tingkat atas yang dikunjungi beberapa kali.

Keuntungan utama dari IDDFS di pohon permainan pencarian adalah bahwa pencarian sebelumnya cenderung meningkatkan heuristik yang umum digunakan, seperti pembunuh heuristik dan alpha-beta, sehingga perkiraan yang lebih akurat dari skor dari berbagai simpul pada pencarian kedalaman akhir dapat terjadi, dan pencarian selesai lebih cepat karena hal itu dilakukan dalam rangka lebih baik. Misalnya, alpha-beta adalah yang paling efisien jika ia mencari langkah terbaik pertama.

Keuntungan kedua adalah respon dari algoritma. Karena iterasi awal menggunakan nilai kecil untuk *d*. Mereka mengeksekusi sangat cepat. Hal ini memungkinkan algoritma untuk memasok indikasi awal hasilnya segera, diikuti oleh perbaikan sebagai *d* meningkatkan. Fasilitas ini memungkinkan program untuk bermain di setiap saat dengan saat ini dalam mencari langkah terbaik yang ditemukan dalam pencarian itu telah menyelesaikan lebih jauh. Hal ini dapat diungkapkan karena setiap kedalaman pencarian rekursif menghasilkan pendekatan yang lebih baik dari solusi, meskipun pekerjaan yang dilakukan pada setiap langkah rekursif. Hal ini tidak mungkin dengan kedalaman-pertama pencarian tradisional, yang tidak menghasilkan hasil antara. Kompleksitas waktu dari IDDFS di pohon yang seimbang berhasil menjadi sama dengan pencarian Depth-first: O(b^d).

III. METODE PENGEMBANGAN

A. *Unified Modelling Language* (UML)

Unified Modelling Language (UML) adalah salah satu alat bantu yang sangat handal di dunia pengembangan sistem yang berorientasi objek. Hal ini disebabkan karena UML menyediakan bahasa pemodelan visual yang memungkinkan bagi pengembang system untuk membuat cetak biru atas visi mereka dalam bentuk yang baku, mudah dimengerti serta

dilengkapi dengan mekanisme yang efektif untuk berbagi (*sharing*) dan mengkomunikasikan rancangan mereka dengan yang lain.

UML merupakan kesatuan dari bahasa pemodelan yang dikembangkan oleh Booch, *Object Modelling 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 desain ke dalam empat tahapan iterative, yaitu : identifikasi kelas-kelas dan obyek-obyek, identifikasi semantik dari hubungan obyek dan kelas tersebut, perincian *interface* dan implementasi.

B. Komponen *UML*

UML mendefinisikan diagram-diagram berikut ini:

1. Use case Diagram menggambarkan fungsionalitas yang diharapkan dari sebuah sistem. Yang ditekankan adalah "apa" yang diperbuat sistem dan bukan "bagaimana". Sebuah use case merepresentasikan sebuah interaksi antara aktor dengan sistem.
2. *Class Diagram*

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

3. Object Diagram

Object diagram serupa den gan diagram kelas, tetapi dari pada menggambarkan kelas objek, lebih baik menggunakan diagram objek yang memodelkan *instance objek actual* den gan menunjukan nilai-nilai saat ini dari *atribute instance.*

4. *Statechart Diagram*

Statechart diagram menggambarkan transisi dan erubahan keadaan (dari satu *state* ke *state* lainnya) suatu objek pada sistem sebagai akibat dari stimuli yang diterima.

5. *Activity Diagram*

Activity diagram menggambarkan berbagai alir aktivitas dalam sistem yang sedang dirancang, bagaimana masing-masing alir berawal, *decision* yang mungkin terjadi, dan bagaimana mereka berakhir. *Activity* diagram juga dapat menggambarkan proses paralel yang mungkin terjadi pada beberapa eksekusi.

6. *Sequence Diagram*

Sequence diagram secara grafis menggambarkan bagaimana objek berinteraksi antara satu sama lain melalui pesan ekusi pada sebuah *usecase* atau operasi.

7. *Collaboration Diagram*

Collaboration diagram juga menggambarkan interaksi antar objek seperrti *sequence diagram*, tetapi lebih menekankan pada p eran masing-masing objek dan bukanp ada waktu penyampaian pesan. Setiap pesan memiliki *sequence number*, di mana pesan dari *level* tertinggi memiliki nomor satu. Pesan dari level yang sama memiliki prefiks yang sama.

8. Variabel pertama adalah bauran pemasaran jasa, yang terdiri dari di-mensi produk (*product*), harga (*price*), lokasi (*place*), promosi (*promotion*), proses (*process*), fasilitas fisik (*physical evidence*) dan orang (sumber daya manusia/*people*).
9. Variabel kedua adalah nilai jasa, yang terdiri dari dimensi manfaat jasa (*service benefit*) dan biaya jasa (*cus-tomer cost*).
10. Variabel ketiga adalah citra perusahaan, yang terdiri dari dimensi *reputation* (seberapa kuat brand perusahaan dikenal mahasiswa), *recognition* (tingginya nilai perusahaan dan persepsi mahasiswa), *affinity* (hubungan emosional yang terjadi antara brand perusahaan dengan mahasiswa), *brand loyalty* (seberapa jauh kesetiaan mahasiswa menggunakan produk/jasa perusahaan).

C. Lazarus

Lazarus adalah platform visual yang gratis dalam *integrated development environment* (IDE) pada *Rapid Application Development* (RAD) yang menggunakan *Free Pascal compiler*, dan mendukung bahasa dari berbagai tingkatan *Object Pascal*. Pengembang perangkat lunak menggunakan Lazarus untuk membuat kode asli konsol dan antarmuka pengguna grafis atau *Graphical User Interface* (GUI) aplikasi untuk desktop, dan juga untuk perangkat *mobile*, aplikasi web, layanan web, komponen visual dan fungsi perpustakaan (.so, .dll, dll, untuk digunakan oleh program lain). *Free Pascal compiler* mendukung sejumlah platform yang berbeda, seperti Mac, Linux dan Windows.

Usaha pertama untuk mengembangkan IDE visual untuk Free Pascal kembali ke tahun 1998, ketika proyek Megido dimulai. Karena berbagai alasan pendekatan ini gagal secara berturut-turut, beberapa pengembang Megido memutuskan untuk memulai sebuah proyek baru berdasarkan landasan yang lebih fleksibel. Versi awal LCL dirilis pada tahun 2001, dan pada tahun 2003 telah diluncurkan versi beta pertama dari Lazarus (0.9.0.3) yang diselenggarakan di *SourceForge* . Dan versi final Lazarus (1.0) dirilis pada tahun 2012. Sementara, pada versi Lazarus 1.2 dengan perangkat tambahan yang signifikan dirilis pada tahun 2014.

Lazarus mewarisi tiga fitur dari penggunaan dari *Free Pascal compiler*, yaitu : mengkompilasi kecepatan, kecepatan eksekusi, dan kompilasi menyilang. Manfaat *compiler Free Pascal* dari struktur bahasa Pascal dan kemajuan yang stabil dari desain Pascal (mencakup beberapa dekade) untuk mengkompilasi aplikasi besar dengan cepat, sering dalam hitungan detik. Ketika kompilasi program referensi kinerja metrik, Lazarus menghasilkan program yang menunjukkan kedekatan atau kinerja yang sama jika dibandingkan dengan program yang sama yang ditulis dalam bahasa C.

Sebuah aplikasi yang dibuat oleh pengembang yang menggunakan Lazarus pada satu platform berpotensi dapat mengkompilasi dan mengeksekusi pada platform Free Pascal yang ada. Petunjuk peringatan biasa dari keterbatasan platform target juga berlaku. Namun, untuk aplikasi desktop satu sumber dapat menargetkan Mac, Linux, dan Windows, biasanya tanpa modifikasi (atau sangat sedikit modifikasi). Contoh aplikasi adalah IDE Lazarus yang itu sendiri diciptakan dengan menggunakan Lazarus IDE dari basis kode tunggal dan tersedia pada semua platform utama dan juga berjalan pada Raspberry PI .

IV. PEMBAHASAN

A. Analisis Permainan *Find The Pair*

Permainan *find the pair* adalah permainan kartu di mana semua kartu diletakkan menghadap ke bawah pada permukaan dan dua kartu dibalikkan menghadap ke atas pada setiap giliran untuk mencari pasangan dari kartu sebelumnya.

Dalam analisis permainan ini dapat mencakup beberapa tugas-tugas sebagai berikut :

1. Membuat tiap kartu memiliki pasangan.
2. Memberikan gambar pada kartu.
3. Mengacak tiap kartu untuk di taruh di permukaan dalam keadaan terbalik.
4. Memperkirakan kartu pertama yang akan di tampilkan dan mencari pasangan dari kartu pertama.

Melakukan penilaian jika kartu memiliki pasangan. Berikut ini cara menganalisa permainan :

1. Aplikasi memproses perkiraan nomor kartu yang akan ditampilkan beserta nomor pasangannya.
2. Pemrosesan pemberian gambar pada tiap kartu (gambar akan sama pada tiap kartu yang berpasangan)
3. Proses pengacakan secara acak sistematik.
4. Proses pemberian gambar latar belakang yang sama pada tiap kartu, sehingga tidak diketahui gambar muka dari tiap kartu.
5. Pemain melakukan pemilihan kartu awal, dan dilanjutkan dengan mencari pasangan yang sama dari kartu tersebut.

Proses akan mendeteksi kartu yang dipilih oleh pemain.

B. Analisis Metode Pengacakan *Systematic Sampling*

Metode pengambilan sampel acak sistematis (*Systematic Random Sampling*) adalah metode pengambilan sampel dengan interval tertentu dari kearangka sampel yang telah ditentukan. Pengambilan sampel secara sistematis adalah suatu metode dimana hanya unsur pertama dari sampel yg dipilih secara acak sedang unsur-unsur selanjut dipilih secara sistematis menurut suatu pola tertentu.

1. Pertama yang harus dilakukan adalah melakukan pendaftaran terhadap seluruh anggota populasi dan diberi nomor secara acak.
2. Tetapkan jumlah anggota kelompok (k) dihitung dengan rumus:

$$k = N/n$$

keterangan :

k = jumlah populasi

N = jumlah bilangan total

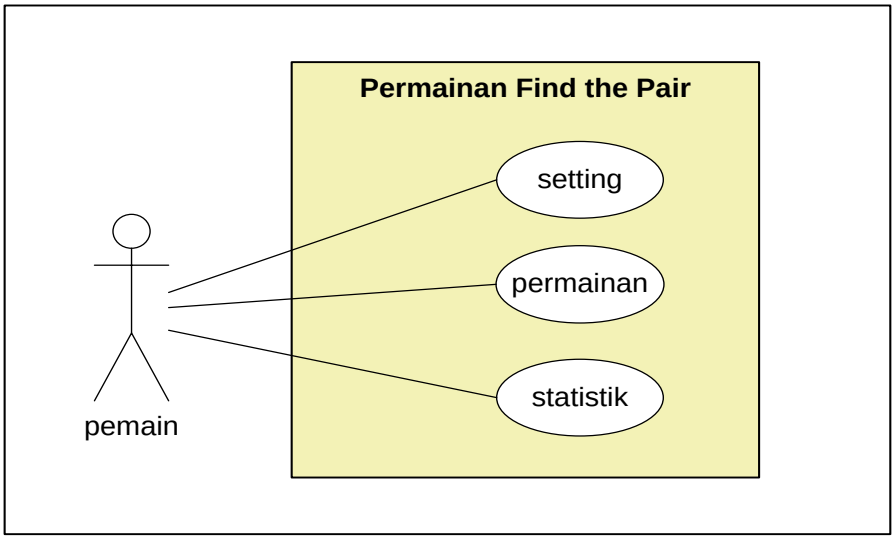
n = jumlah bilangan interval

V. Perancangan Sistem

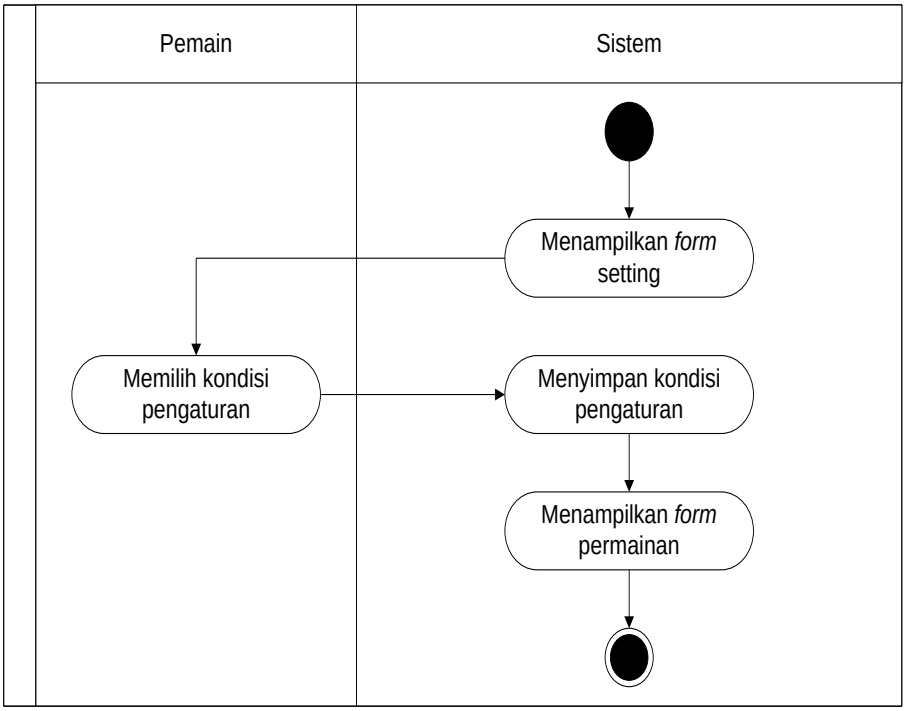
A. *Use Case Diagram*

Sebuah model harus berfokus pada kebutuhan yang terlihat tanpa masalah atau domain bisnis dan ditulis sebagai level yang relatif tinggi. Pemodelan use case adalah pemodelan sistem dari perspektif pandangan pemakai akhir (end user). Model use case adalah pandangan

dari luar sistem, sementara model rancangan adalah pandangan dari dalam. Model use case menangkap penggunaan-penggunaan sistem, sedangkan model rancangan merepresentasikan pembangunan dari sistem.

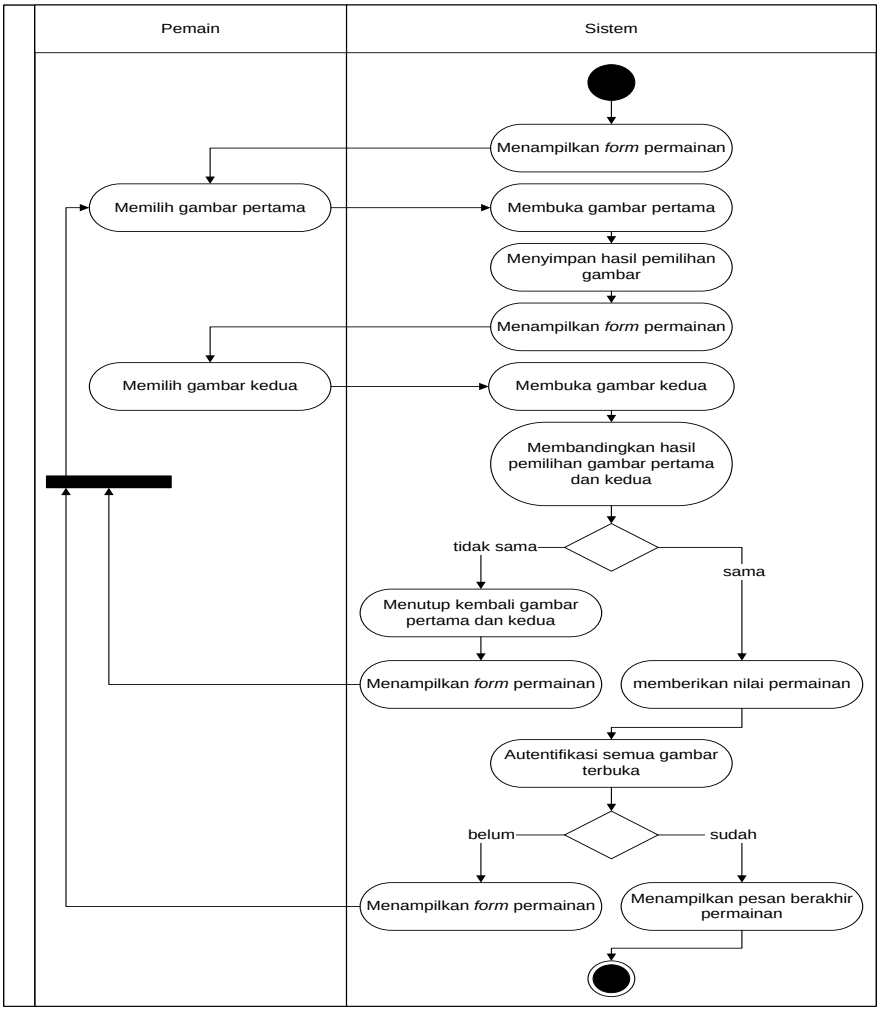


B. *Activity Diagram*

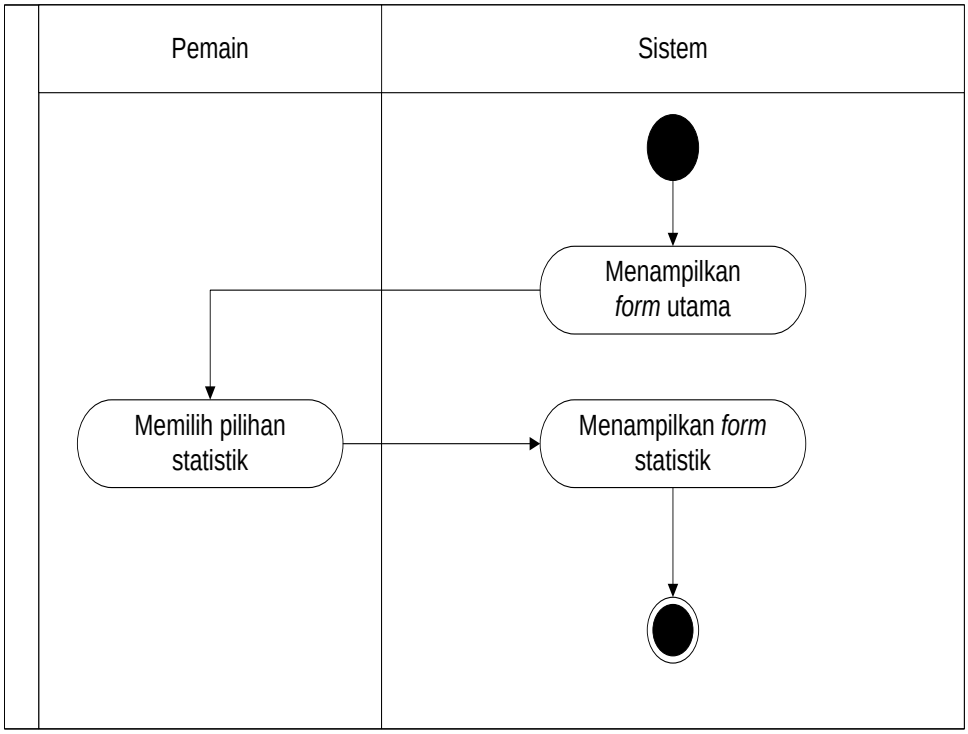


1. Sistem menampilkan *form* permainan yang akan dimainkan oleh pemain.
2. Pemain memilih gambar pertama yang tertutup.
3. Sistem akan membuka gambar pertama yang dipilih pemain.
4. Selanjutnya, menyimpan *index* dari gambar pertama.
5. Sistem melanjutkan menampilkan *form* permainan.
6. Pemain memilih gambar kedua yang tertutup.
7. Sistem membuka gambar kedua yang telah dipilih pemain.
8. Lalu, sistem akan membandingkan nilai *index* dari gambar kedua yang dipilih dengan *index* gambar pertama sebelumnya.
9. Ketika hasil banding antara gambar pertama dan kedua tidak sesuai, maka sistem akan menutup kembali gambar pertama dan gambar kedua, dan kembali menampilkan *form* permainan, dan dilanjutkan dengan keterangan nomor 2 (dua).
10. Ketika perbandingan *index* gambar pertama dan kedua sesuai, maka adanya penambahan nilai permainan.
11. Selanjutnya, sistem akan mendeteksi apakah semua gambar telah terbuka atau belum.
12. Ketika semua gambar belum terbuka, sistem kembali menampilkan *form* permainan, dan dilanjutkan kembali seperti keterangan nomor 2 (dua).
13. Permainan berakhir ketika semua gambar telah terbuka.

C. *Activity Diagram* pada Permainan

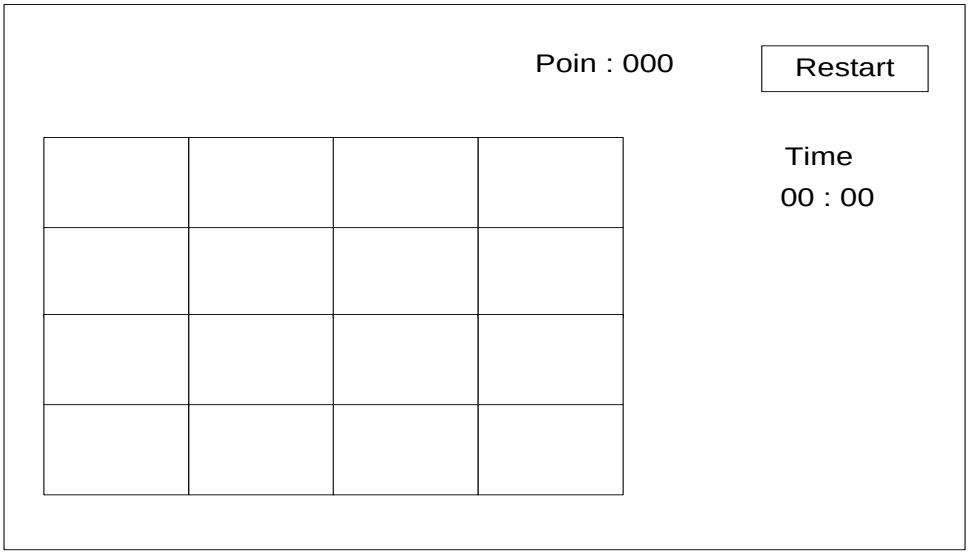


D. *Activity Diagram* pada *Statistic* Permainan



E. Rancangan Input

Perancangan input merupakan tampilan dirancang sedemikian rupa agar mudah dioperasikan oleh pemakai. Untuk setiap eksekusi peralatan yang digunakan berupa *keyboard* dan *mouse*. Berikut gambaran dari *form* permainan yang penyusun rancang dalam membangun aplikasi ini :



Gambar Perancangan *form* permainan

IV. KESIMPULAN DAN SARAN

A. Kesimpulan

1. Penerapan unsur Indonesia, yaitu berupa gambar tokoh pahlawan dari Indonesia, dan juga menyangkut kebudayaan Indonesia, seperti alat music daerah Indonesia, candi-candi dari berbagai tempat yang ada di Indonesia, dan juga pakaian adat yang dikenakan dari berbagai suku yang ada di Indonesia.
2. Penelitian ini telah menerapkan metode pencarian yang berdasarkan *iterative deepening search.*
3. Menerapkan sistem penilaian permainan dengan skala bertingkat atau dikenal dengan sebutan *rating scale.*
4. Pengimplementasian proses metode pengacakan secara sistematis, atau biasa disebut dengan teknik pengambilan *sample* secara *systematic*.

B. Saran dan Rekomendasi

1. Pengimplementasian permainan yang menggunakan metode lain selain *iterative deepening search*.
2. Penerapan proses permainan yang bisa dimainkan dengan pengguna lain, sehingga bisa menambah tingkat kesenangan pengguna.
3. Ditambahnya fitur *upload* gambar untuk tema, sehingga pengguna dapat memasukkan gambar kesukaan mereka ke dalam permainan.
4. Penambahan fitur *auto maximize* sehingga pengguna dapat memainkannya dengan lebih leluasa.

DAFTAR PUSTAKA

Buku Panduan. *Buku panduan untuk Laporan Kerja Praktek dan Skripsi untuk mahasiswa Sekolah Tinggi Teknologi Indonesia Tanjungpinang*. 2014

Depdiknas. *Kamus Besar Bahasa Indonesia Pusat Bahasa Edisi Keempat,* 2008, Jakarta : PT. Gramedia Pustaka Utama

Entertainment Software Association. *Essential Facts about The Computer and Video Game Industry*. 2013

Korf, Richard. "*Depth-first Iterative-Deepening: An Optimal Admissible Tree Search*". Artificial Intelligence 27: 97–109, 1985.

S. Pressman, Roger. *Rekayasa Perangkat Lunak*, 2002, ANDI Yogyakarta.

Verdi Yasin.S.Kom.,M.Kom. *Rekayasa Perangkat Lunak Berorientasi Objek*,2012,Mitrawacanamedia.

Reviere, Rebecca. *Needs Assessment: A Creative and Practical Guide for Social Scienties*, 1996, Taylor&Francis

Russell, Stuart J.; Norvig, Peter, *Artificial Intelligence: A Modern Approach,* 2003

Kamus Bahasa Indonesia Online. [Online] Available: http://kamusbahasaindonesia.org/permainan. [2016, Mei 22]
Concentration Game. [Online] Available: https://en.wikipedia.org/wiki/Concentration_(game). [2016, Mei 22]
Samuel Henry. *Panduan Praktis Membuat Game 3D.* 2005, Graha Ilmu.
Lazarus. [Online] Available: http://www.lazarus-ide.org. [2016, Mei 15]
Lazarus (IDE). [Online] Available: https://en.wikipedia.org/wiki/Lazarus_(IDE). [2016, Juni 15]
Microsoft Visio. [Online] Available: https://id.wikipedia.org/wiki/Microsoft_Visio. [2016, Agustus 2]
Pengertian Software (perangkat lunak) Komputer. [Online] Available: http://belajar-komputer-mu.com/pengertian-software-perangkat-lunak-komputer. [2016, Mei 22]