1. **LATAR BELAKANG**

Pada zaman sekarang ini, terdapat banyak sekali jenis-jenis game. Dari *game* berbasis *desktop*, *mobile* sampai *game online*. *Game* sendiri banyak yang meminatinya, dari usia anak-anak hingga dewasa. Salah satu contohnya adalah sebuah *game* sederhana yang disebut *memory match game.*

*Memory match game* merupakan permainan pencocokan gambar yang memerlukan ketajaman ingatan. Permainan ini dimainkan oleh satu pemain. Cara bermain permainan ini cukup mudah, pemain hanya perlu menekan kartu-kartu (*panel*/kotak) yang masih tertutup, lalu gambar akan muncul. Setelah itu, mencari pasangannya pada kartu-kartu lain yang masih tertutup. Jika bukan sepasang, maka kartu akan tertutup kembali, dan jika sepasang, maka kedua kartu akan terus terbuka. Pemain diberikan waktu untuk menyelesaikan permainan ini, dan jika waktu penyelesaiannya lebih cepat dari pemain lain, maka namanya akan tersimpan di *high score.*

PBO (Pemrograman Berorientasi Objek) adalah suatu konsep pemrograman yang memanfaatkan kelas dan objek untuk keperluan penulisan program. Salah satu contoh bahasa pemrograman dengan orientasi objek adalah *Java*. Lebih khususnya, ada aplikasi bernama *Netbeans* yang dapat digunakan untuk membuat suatu program *Java* dengan basis GUI (*Graphical User Interface*). Aplikasi ini dapat mempermudah *developer* untuk mengembangkan berbagai jenis program *Java* dengan basis GUI.

Karena itulah, di sini akan dibuat suatu program *Java* bernama *Memory Match Game*, dengan memanfaatkan segala fitur yang ada dalam aplikasi *Netbeans.*

1. **PERANCANGAN SISTEM**
2. *Entity Relationship Diagram* (ERD)

|  |
| --- |
|  |

**Gambar 1.** ERD *Memory Match Game*

**Gambar 1** merupakan ERD dari *game* *Memory Match*. ERD tersebut memiliki dua entitas, yaitu entitas pemain (atribut: id\_pemain, nama\_pemain, dan skor) yang berguna untuk menyimpan data skor pemain, dan entitas gambar (atribut: id\_gambar dan nama\_gambar) yang berfungsi sebagai penyimpan gambar. ERD ini tidak memiliki relasi, karena di antara dua entitas tersebut tidak ada keterkaitan satu sama lain.

1. *Use Case Diagram*

|  |
| --- |
|  |

**Gambar 2.** *Use Case Diagram* aktor pemain

**Gambar 2** merupakan *use case diagram* dari sisi pemain. Dimana *use case* ini berisi informasi mengenai hal-hal yang dapat dilakukan pemain.

Hal-hal yang dapat dilakukan *user*, yaitu:

* Memasukkan nama
* Melihat menu permainan
* Melihat nilai terbaik
* Keluar dari permainan
* Memainkan *game* pencocokan gambar

1. *Class Diagram*

|  |
| --- |
|  |

**Gambar 3.** *Class diagram*

Dalam pembuatan program *Memory Match Game* ini, akan digunakan 5 kelas, yaitu:

1. Kelas Utama

Di dalam kelas ini akan diisi *method main* yang akan memulai program.

1. Kelas Tampilan

Kelas ini berguna sebagai pembuatan tampilan secara keseluruhan dari program *Memory Match Game*. Dimana dalam kelas ini terdapat beberapa atribut *public jLabel*, *jButton*, *icon*, dan *jPanel*. Lalu, *method* yang ada dalam kelas ini adalah *method*  konstruktor Tampilan, *run*(), dan *mouseClicked*().

1. Kelas Gambar

Kelas ini berguna untuk mendapatkan *id* dan *path directory* dari gambar yang akan dimuat. Lalu gambar akan dimuat sesuai dengan *path* yang telah ditentukan.

1. Kelas Pemain

Kelas Pemain ini digunakan untuk menyimpan data pemain beserta skor mereka dalam *database* sekaligus mendapatkan nama pemain saat *method getNama*() dijalankan.

1. Kelas Tampil\_Skor

Pada kelas ini, terdapat atribut *public tSkor* dan *method showSkor*() yang akan digunakan untuk menampilkan skor-skor tertinggi dari pemain.

1. **RANCANGAN PROGRAM**

Permainan *Memory Match* merupakan permainan yang terdiri dari satu pemain atau *single player*. Permainan *Memory Match* ini mengutamakan kecepatan dan ketelitian dalam bermain, inti utama permainan ini dalah pencocokan gambar.

Prosedur dari program yang akan dibuat dapat diuraikan sebagai berikut:

1. Pemain memasukkan nama mereka saat *window game* pertama kali muncul, program akan menyimpan nama mereka dalam *database* bersamaan dengan skor yang mereka dapatkan.
2. Program akan menampilkan menu utama dari program *Memory Match Game*, yang terdiri atas 3 pilihan: *start new game* untuk memulai *game* baru, *high score* untuk melihat daftar pemain dengan skor tertinggi, dan *exit* untuk keluar dari program.
3. Jika pemain menekan pilihan *start new game*, maka akan tampil *window* baru yang menjadi tujuan utama dari program ini, yaitu permainan *Memory Match. Path* dari masing-masing pasangan gambar akan diperoleh dari *database*, kemudian posisi diacak dengan menghadap bawah (agar tersembunyi dari pemain).
4. Saat pemain menekan suatu kartu (*panel*/kotak yang berbentuk kartu), maka program akan menunjukkan gambar yang ada di baliknya. Lalu, pemain diharuskan menekan satu kartu lagi. Jika gambar pada kartu sama dengan kartu yang dibuka sebelumnya, maka kedua kartu akan terus terbuka. Jika tidak, maka kedua kartu kembali tertutup. Begitu seterusnya hingga semua kartu dalam *board game* terbuka. Waktu dihitung sejak saat pemain menekan pilihan *start new game* hingga semua kartu terbuka.
5. Jika pemain menekan pilihan *high score*, maka program akan menampilkan *window* yang di dalamnya berisi beberapa nama dan skor pemain tertinggi. Data nama dan skor ini dipanggil dari *database.*
6. Jika pemain menekan pilihan *exit*, maka program akan terhenti.
7. **RANCANGAN GUI PROGRAM**

Berikut ini adalah *user interface* awal dalam pengembangan program *Memory Match Game*:



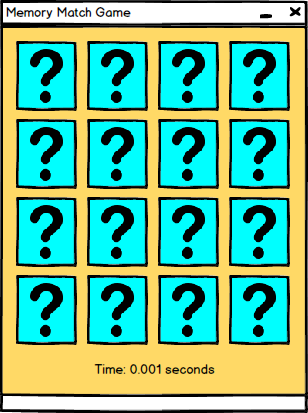
**Gambar 4.** *Window* untuk *input* nama

*Window* pada gambar di atas adalah tampilan yang akan muncul pertama kali. Di sini pemain dapat mengisi nama mereka, lalu menekan tombol masuk untuk sampai ke menu utama.



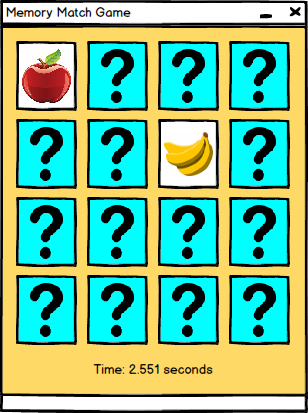
**Gambar 5.** Tampilan menu utama

Tampilan pada **gambar 5** adalahh menu utama dari program *Memory Match Game* dimana pemain dapat memilih beberapa menu yang ada.



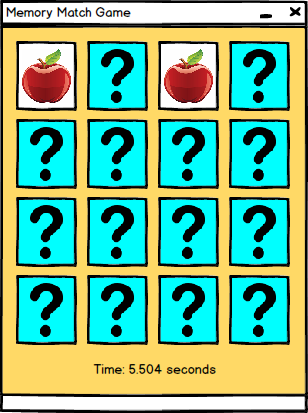
**Gambar 6.** Tampilan *in-game Memory Match Game*

Pada **gambar 6** adalah tampilan yang akan muncul saat pemain menekan pilihan *start new game*. Di sinilah permainan dilakukan, pemain harus bisa membuka semua kartu yang ada dalam *board.*



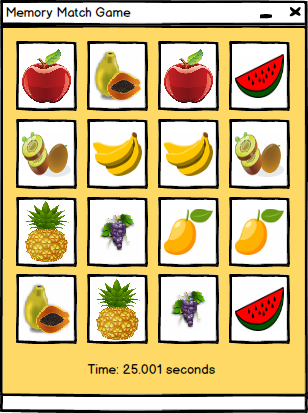
**Gambar 7.** Tampilan *in-game Memory Match Game*

Tampilan pada gambar di atas adalah contoh saat pemain membuka dua kartu yang bukan sepasang gambar. Karenanya kartu akan tertutup kembali.



**Gambar 8.** Tampilan *in-game Memory Match Game*

Tampilan pada gambar di atas adalah contoh saat pemain membuka dua kartu yang merupakan sepasang gambar. Karenanya kedua kartu akan terus terbuka.



**Gambar 9.** Tampilan *in-game Memory Match Game*

Tampilan di atas adalah contoh jika seluruh kartu sudah terbuka. Waktu yang dibutuhkan pemain untuk menyelesaikan permainan juga akan muncul di bagian bawah.



**Gambar 10.** Tampilan menu *high score*

*Window* di atas akan tampil jika pemain memilih menu *high score* saat berada di menu utama. Di sini akan terlihat nama pemain dan skor mereka. Nama dan skor yang ditampilkan hanyalah pemain dengan waktu tercepat.