

**LAPORAN PROYEK PRAKTIKUM PEMROGRAMAN
BERORIENTASI OBJEK 2022
PENGIMPLEMENTASIAN
PEMROGRAMAN BERORIENTASI OBJEK
PADA PERMAINAN ROCKET ASTEROID (OCID)**



Oleh Kelompok 2B

Anggota:

- | | | |
|----------------------------------|-------------|-----|
| 1. Mahendra Putra Raharja | (F1D020047) | (B) |
| 2. Nur Rahmah Utami Putri | (F1D020075) | (B) |
| 3. Muhammad Syafa Asgani | (F1D020065) | (B) |
| 4. Muhammad Lanang Agung Anggoro | (F1D020059) | (B) |
| 5. Rizwan Alfian | (F1D020091) | (A) |
| 6. Riski Darmawan | (F1D020089) | (A) |
| 7. Agus Gunawan | (F1D020002) | (A) |

**PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS MATARAM
2022**

LEMBAR PENGESAHAN LAPORAN PROYEK

PRAKTIKUM PEMROGRAMAN BERORIENTASI OBJEK 2022

1. Kelompok : 2B
2. Judul Proyek : Pengimplementasian Pemrograman Berorientasi Objek
pada Permainan *Rocket Asteroid* (OCID)
3. Anggota Kelompok : Mahendra Putra Raharja (F1D020047)
Nur Rahmah Utami Putri (F1D020075)
Muhammad Lanang Agung Anggoro (F1D020059)
Muhammad Syafa Asgani (F1D020065)
Rizwan Alfian (F1D020091)
Riski Darmawan (F1D020089)
Agus Gunawan (F1D020002)

Laporan proyek ini disusun sesuai dengan kaidah penyusunan yang telah ditentukan dan dibuat sebagai syarat mata kuliah pemrograman berorientasi objek 2022.

Mataram, 16 Desember 2022

Telah diperiksa dan disahkan oleh:

Koordinator Asisten

Asisten Pembimbing

Muhammad Ari Rifqi
F1D019068

Dafa Rozzi Pratama
F1D019016

Dalam permainan OCID ini terdapat beberapa *class* yang telah diimplementasikan ke dalam bentuk *source code* yaitu:

a. *Class Entity*

Class Entity ini digunakan untuk mengatur entitas yang digunakan saat *game* dimainkan. Adapun *method* yang digunakan dalam *class* ini adalah sebagai berikut:

1. *Entity()* : merupakan *constructor* yang digunakan untuk melakukan *setting* terhadap titik koordinat dari sebuah entitas saat berada di lintasan luar angkasa/permainan.
2. *getPosX()* : Merupakan *accessesor* yang digunakan untuk mengambil nilai integer yang merupakan titik koordinat x dari sebuah entitas saat di luar angkasa/lintasan.
3. *getPosY()* : Merupakan *accessesor* yang digunakan untuk mengambil nilai *integer* yang merupakan titik koordinat y dari sebuah entitas saat di luar angkasa/lintasan..
4. *setPosX()* : Merupakan *mutator* yang digunakan untuk memberikan nilai *integer* terbaru yang merupakan titik koordinat x terbaru dari sebuah entitas saat di luar angkasa/lintasan.
5. *setPosY()* : merupakan *mutator* yang digunakan untuk memberikan nilai *integer* terbaru yang merupakan titik koordinat yg terbaru dari sebuah entitas saat di luar angkasa/lintasan.
6. *moveLeft()* : merupakan *method* yang digunakan untuk merubah posisi entitas menjadi lebih geser ke sebelah kiri atau secara sistem merubah nilai x yang merupakan posisi awal entitas menjadi lebih kurang dari sebelumnya agar posisi koordinat x entitas bergeser ke kiri.
7. *moveRight()* : Merupakan *method* yang digunakan untuk merubah posisi entitas menjadi lebih geser ke sebelah kanan atau secara sistem merubah nilai x yang merupakan posisi awal entitas menjadi bertambah dari sebelumnya agar posisi koordinat x entitas bergeser ke kanan.

b. *Class Traffic*

Class Traffic berfungsi sebagai *class* yang melakukan *multi-threading*, yaitu untuk dapat menjalankan *task* atau tugas lebih dari satu secara paralel, sehingga dengan *class* ini *task* yang banyak akan cepat selesai karena tidak saling tunggu untuk menyelesaikan *task*.

c. *Class Game*

Class Game digunakan untuk mengatur jalannya permainan yang dimulai dari pemilihan menu, penampilan peringkat, permainan dimulai hingga keluar dari permainan. Berikut *method* yang digunakan dalam *class* ini.

1. *Game()* : merupakan *constructor* yang digunakan untuk menginisialisasikan seluruh objek yang ada pada *class* ini. Objek yang dimaksud di antaranya adalah objek *database*, *state game*, ukuran jendela, *asteroid*, hingga memulakan *traffic* atau lintasan permainan.
2. *restartGame()* : Merupakan *method* yang digunakan untuk *setting* ulang permainan setelah permainan sebelumnya telah berakhir agar dapat dimainkan kembali.
3. *keyPressed(KeyEvent event)* : Merupakan *method* yang digunakan untuk mengatur hasil *input*-an *user* yang diberikan saat mengklik beberapa *key press* pada *keyboard*. Beberapa *key press* yang diatur dalam *method* ini adalah tombol bawah dan atas yang digunakan *user* di dalam memilih menu, tombol *enter* saat *user* memilih salah satu menu atau pilihan pada halaman menu, tombol kiri dan kanan saat *user* memainkan permainan dan mulai memindahkan posisi *rocket* dari posisi sebelumnya serta tombol *backspace* saat *user* ingin kembali ke halaman menu utama setelah melihat klasemen sementara permainan.
4. *paint(Graphics g)* : Merupakan *method* yang digunakan untuk menampilkan konten pada halaman menu dan permainan.

d. *Class Setting*

Class Setting ini digunakan untuk mengatur sistem permainan secara umum seperti *background* menu, lintasan permainan, ukuran *window*, dan *default* posisi *rocket*.

e. *Class Window*

Selain daripada *class* tersebut terdapat beberapa *class* yang lain yaitu *Thread*, *JPanel*, <interface> *KeyListener*, dan *JFrame* yang merupakan *class library* untuk membantu pembuatan permainan OCID ini.

f. *Class Music*

Class Music merupakan *abstract class* yang di dalamnya terdapat beberapa *method abstract* yaitu *PlayMusicMenu()*, *PlayMusicKlik()* dan *PlayMusicStart()*, *PlayMusicGameOver()*.

g. *Class PlayMusic*

Class ini merupakan *subclass* dari *class* Music yang mana pada *class* PlayMusic ini akan melakukan *override* terhadap *method* pada *class* Music sebagai berikut.

1. PlayMusicMenu(): *Method* ini berfungsi untuk mengatur musik yang akan dimainkan pada menu permainan.
2. PlayMusicKlik(): *Method* ini berfungsi untuk mengatur musik/suara yang akan dimainkan ketika melakukan klik atau memilih salah satu menu permainan.
3. PlayMusicStart(): *Method* ini berfungsi untuk mengatur musik yang akan dimainkan pada saat permainan dimulai.
4. PlayMusicGameOver(): *Method* ini berfungsi untuk mengatur musik yang akan dimainkan pada saat permainan selesai atau mengalami kekalahan.

h. *Class Database*

Class Database digunakan untuk melakukan koneksi program java yang dibuat dengan *database*. Koneksi ini dilakukan agar program dapat menyimpan datanya ke *database* dan menampilkannya pada halaman permainan. Terdapat sebuah *constructor* dalam *class* ini yang digunakan untuk melakukan koneksi dengan *database* dari *driver* hingga melakukan koneksinya langsung.

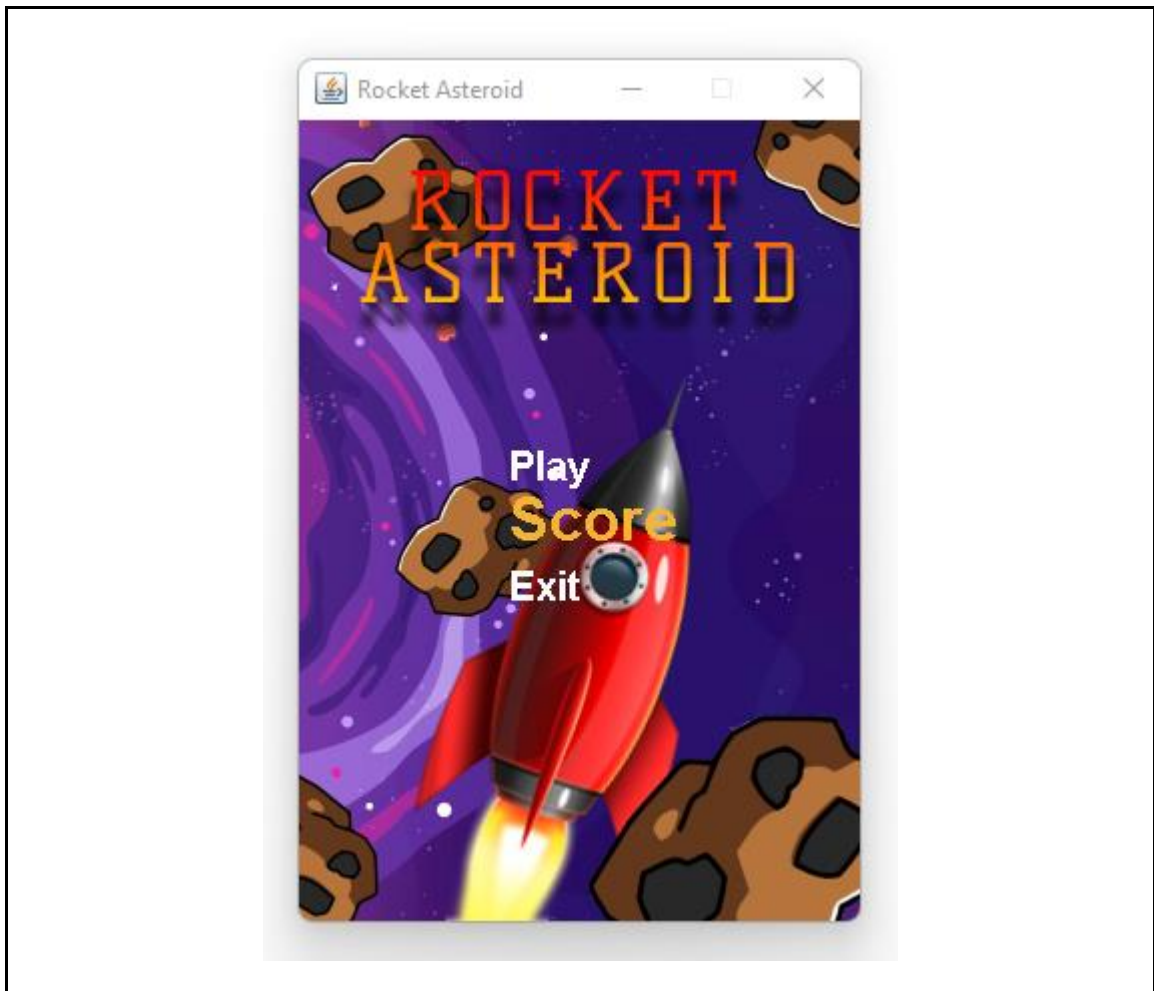
i. *Class Rocket*

Class Rocket merupakan *class extends* atau turunan dari *class entity*. Di mana pada *class* ini terdapat konstraktor *Rocket* dengan 2 parameter yang berasal dari *class* induknya. Selain itu terdapat *method* draw() dengan 1 parameter dan merupakan *method* yang meng-*input*-kan dan mengatur objek *rocket* yang terdapat dalam permainan.

j. *Class Asteroid*

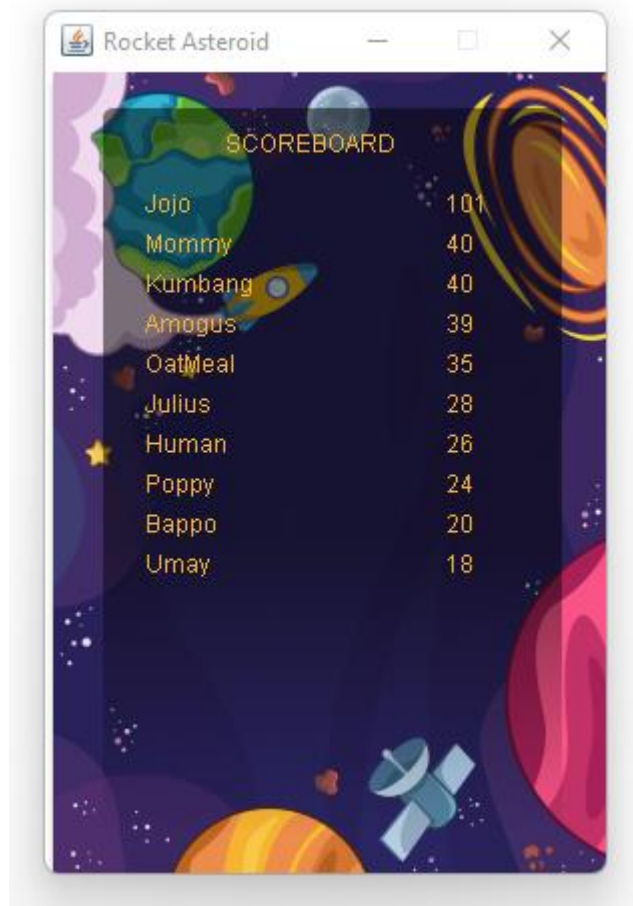
Class Asteroid merupakan turunan atau *extends* dari *class Entity*. Pada *class* ini, terdapat *method* draw() dan *method* drawImage() yang masing-masing berfungsi untuk mengambil gambar/*image background* dan *asteroid* pada *assets* dan mengaplikasikannya ke dalam *window/GUI* permainan dan mengaplikasikan panjang dan lebar gambar menyesuaikan *window* dari permainan.

5. Tampilan GUI



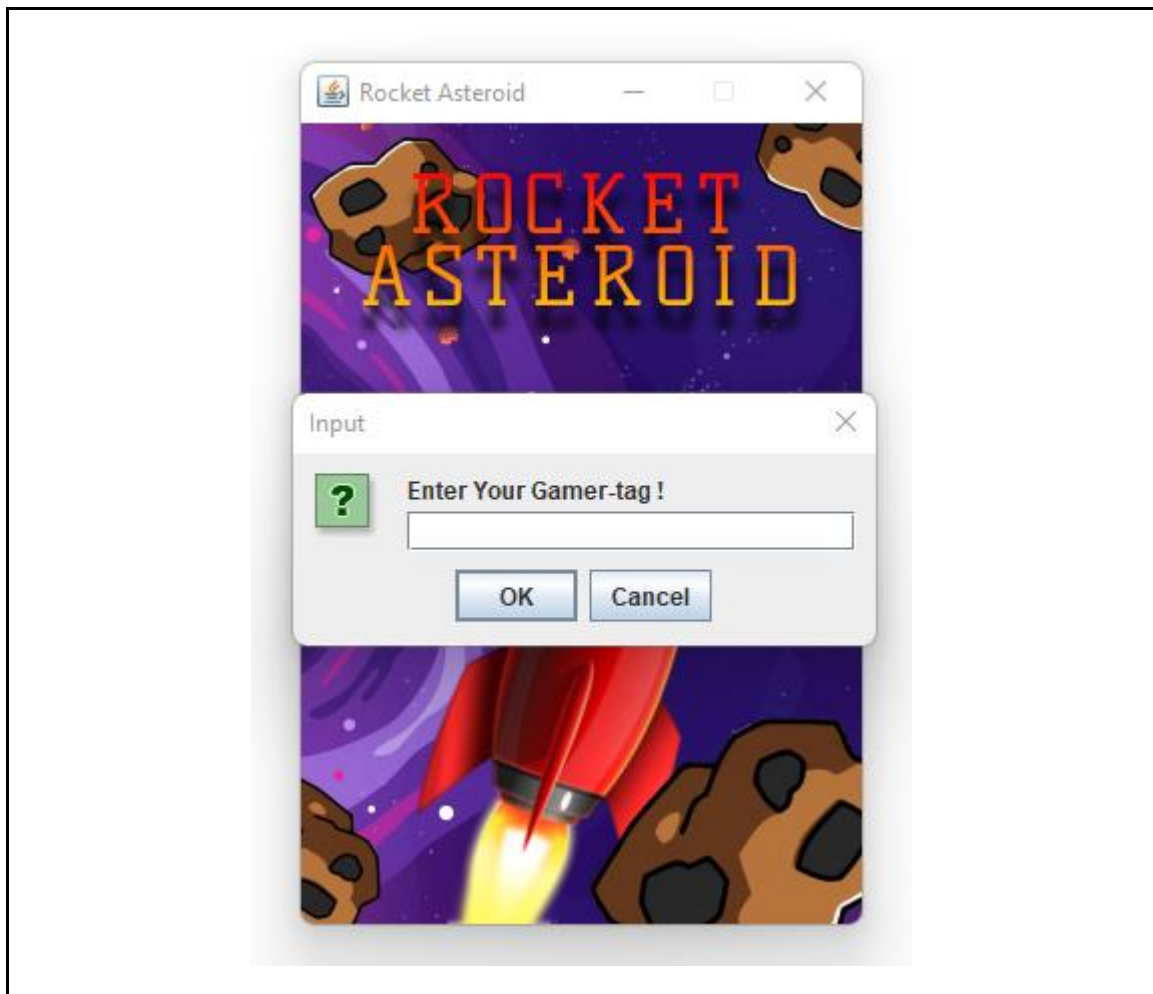
Gambar 1 Main Menu

Pada Gambar Main Menu disini merupakan tampilan atau GUI awal dari *game Rocket Asteroid* ketika awal dibuka. Di mana di dalamnya terdapat menu *play* untuk memulai permainan, menu *Score* untuk melihat histori *score* dari *game* yang sudah di mainkan, dan menu *exit* untuk keluar dari *game*.



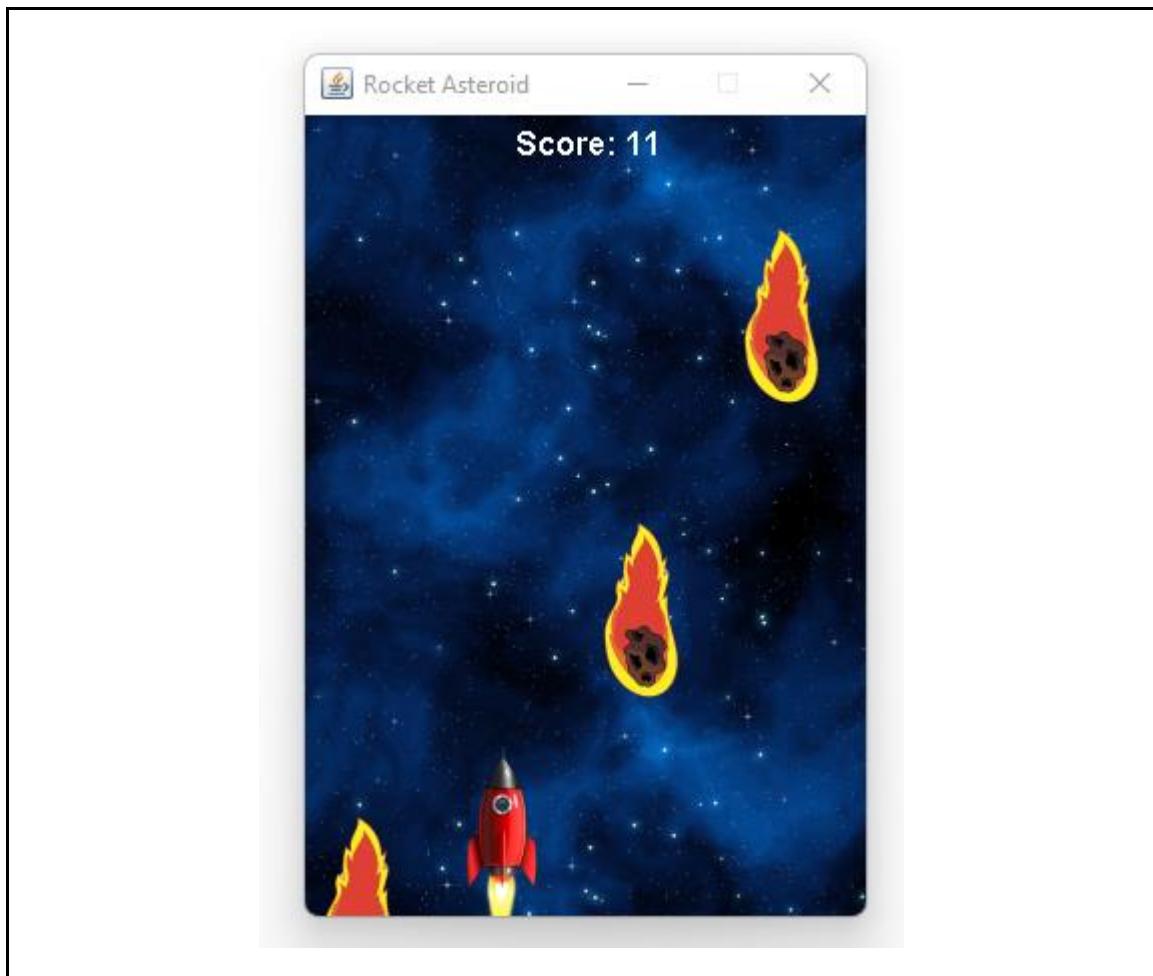
Gambar 2 *Scoreboard*

Gambar *Scoreboard* di atas merupakan tampilan dari menu *scoreboard*, dimana di dalamnya terdapat *history score* dan nama *user* yang telah memainkan *game*, dan skornya di urutkan dari *score* tertinggi ke *score* terendah.



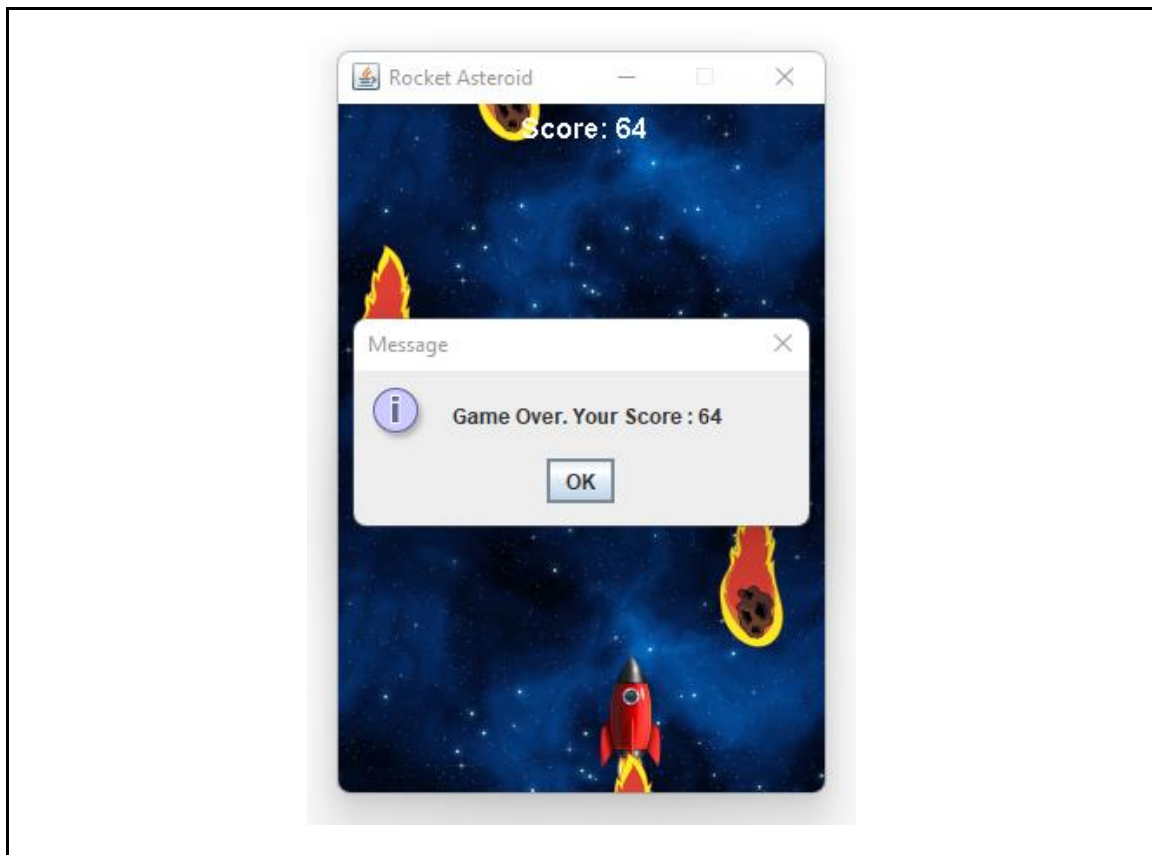
Gambar 3 *Input Nama Player*

Gambar *Input Nama Player* di atas ini merupakan tampilan *menu* untuk memasukan nama *user* atau pemain, dimana di dalamnya terdapat kolom tempat memasukan nama, *button Cancel* untuk membatalkan penggunaan nama yang telah di *input*, dan *button OK* untuk masuk ke game dan memverifikasi penggunaan nama yang telah di *-input-*kan. Menu *Input Nama Player* ini muncul setelah mengklik *menu Play* pada tampilan awal.



Gambar 4 *Gameplay*

Gambar *Gameplay* di atas adalah tampilan dari *game* yang sedang dimainkan, dimana disini *player* harus menghindari *asteroid*, *score* bertambah berdasarkan jumlah *asteroid* yang muncul pada permainan yang sedang berlangsung. Semakin tinggi skor yang didapatkan pemain maka gerakan *asteroid* semakin cepat, terjadi peningkatan kesulitan pada *game*.



Gambar 5 *Game Over*

Gambar 5 *Game Over* di atas adalah *alert* pada halaman permainan yang di mana akan ditampilkan ketika *rocket* telah menabrak *asteroid* atau dapat dikatakan pemain kalah. *Alert* tersebut berisi pesan *Game Over* dan disertai skor yang berhasil didapatkan pemain. Skor terakhir yang berhasil didapatkan pemain akan disimpan dalam *database*.

6. Link Github

Berikut merupakan link Github untuk mengakses *source code* permainan *Rocket Asteroid* (OCID):

[tubespbo2022/Kelompok-B/ProjectPraktikumFinal/src](https://github.com/tubespbo2022/Kelompok-B/ProjectPraktikumFinal/src_dafarozzi16/tubespbo2022) at [main](#)
[dafarozzi16/tubespbo2022 \(github.com\)](https://github.com/dafarozzi16/tubespbo2022)

7. Referensi

Terdapat pula referensi pada permainan *Rocket Asteroid* (OCID) ini sebagai berikut :

- [1] M. Ahyar, “Game Simulasi Desain Model Dan Modifikasi Mobil Dua Dimensi Berbasis Android,” *Ubiquitous Comput. its Appl. J.*, vol. 2, hal. 39–44, 2019, doi: 10.51804/ucaiaj.v2i1.39-44.
- [2] Tutorial Greenfoot Part 4 – Menyisipkan Backsound pada Game February 22, 2021 by Semar Soft
- [3] <https://github.com/Ajaysinh1290/Car-Racing-game-in-java>