Jurnal Sistem Dan Teknologi Informasi (JSTI)

https://journalversa.com/s/index.php/jsti Vol. 07, No. 2, Mei 2025

IMPLEMENTASI KONSEP PEMROGRAMAN BERORIENTASI OBJEK PADA PENGEMBANGAN GAME KUIS SEDERHANA MENGGUNAKAN C# UNITY

Dheany Angelina Putri Sembiring[1]
Email: angelinadheany@gmail.com

Muhammad Zaki Ulwi[2]
Email: mz057545@gmail.com

Erika togito Siahaan[3]
Email: erikatogitoshn@gmail.com

Muhammad Fakhri Lubis[4]
Email: fakhrilbs123@gmail.com

[1,2,3,4]Universitas Negeri Medan

ABSTRAK

Penelitian ini bertujuan untuk mengimplementasikan konsep Pemrograman Berorientasi Objek (OOP) dalam pengembangan game kuis interaktif sederhana menggunakan game engine Unity. Penelitian ini menggunakan metode deskriptif untuk menjelaskan proses pengembangan serta penerapan prinsip OOP, seperti enkapsulasi, pewarisan, polimorfisme, dan abstraksi, dalam sistem game. Aplikasi dikembangkan menggunakan bahasa pemrograman C# dengan memanfaatkan dukungan Unity terhadap paradigma OOP untuk membentuk kode yang modular dan dapat digunakan kembali. Hasil menunjukkan bahwa game berjalan secara efektif dengan antarmuka yang ramah pengguna dan alur permainan yang fungsional. Penelitian lebih lanjut diperlukan untuk mengembangkan game ini secara lebih mendalam, terutama dalam hal penggunaan paradigma pemrograman yang lebih kompleks dan peningkatan efisiensi sistem.

Kata Kunci: Pemrograman Berbasis Objek, Game Kuis Interaktif, C#, Unity.

ABSTRACT

This study aims to implement the concept of Object-Oriented Programming (OOP) in the development of a simple interactive quiz game using the Unity game engine. This study uses a descriptive method to explain the development process and the application of OOP principles, such as encapsulation, inheritance, polymorphism, and abstraction, in the

game system. The application was developed using the C# programming language by utilizing Unity's support for the OOP paradigm to form modular and reusable code. The results show that the game runs effectively with a user-friendly interface and functional gameplay. Further research is needed to develop this game in more depth, especially in terms of using more complex programming paradigms and increasing system efficiency.

Keywords: *Object-Oriented Programming, Interactive Quiz Game, C#, Unity.*

1. PENDAHULUAN

Pada era digital saat ini, inovasi pada bidang teknologi pemrograman telah berkembang sangat pesat sehingga mendorong pengembangan perangkat lunak yang canggih dan efisien. Salah satu paradigma pemrograman yang banyak diterapkan adalah Pemrograman Berorientasi Objek (OOP). Pemrograman Berorientasi Objek (OOP) adalah paradigma pemrograman yang berorientasi pada objek, di mana program disusun berdasarkan objek yang memiliki atribut (data) dan metode (fungsi). Paradigma ini dirancang untuk meningkatkan modularitas, fleksibilitas, serta kemudahan dalam pemeliharaan kode melalui empat prinsip utama: enkapsulasi, pewarisan, polimorfisme, dan abstraksi (Nagineni, 2021). Dengan pendekatan ini, OOP memungkinkan pengembang perangkat lunak untuk lebih mudah merancang, mengelola, dan mengembangkan aplikasi dengan struktur yang lebih terorganisir (Savinov, 2008).

Dalam penelitian ini, kami mengimplementasikan konsep Pemrograman Berorientasi Objek (OOP) untuk mengembangkan game kuis sederhana menggunakan game engine Unity. Unity sendiri merupakan platform yang banyak digunakan untuk membuat aplikasi interaktif, mulai dari game 2D dan 3D, simulasi, hingga visualisasi berbasis augmented reality dan virtual reality (Hussain et al., 2020). Salah satu alasan utama pemilihan Unity adalah karena Unity mendukung terhadap pemrograman berbasis objek (OOP) melalui bahasa C#, yang memudahkan pengembang dalam membangun sistem game yang lebih modular, fleksibel, dan mudah dikembangkan (Nieminen, 2021).

2. METODE PENELITIAN

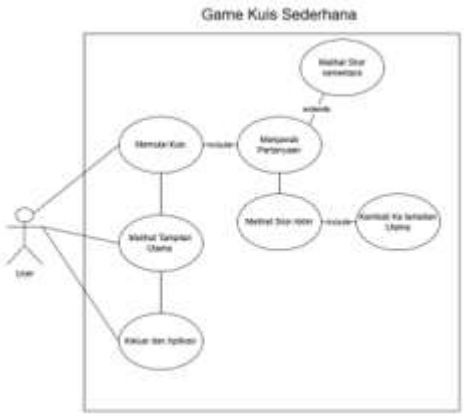
Penelitian ini menggunakan metode deskriptif. Menurut Prastowo & Danianti (2023), metode deskriptif adalah pendekatan yang sering digunakan dalam pengembangan perangkat lunak untuk menganalisis proses pengembangan secara terstruktur dan sistematis. Penelitian ini bertujuan untuk memberikan gambaran

sistematis mengenai penerapan konsep Pemrograman Berorientasi Objek (PBO) dalam pengembangan game kuis sederhana berbasis C# dan Unity.

3. HASIL DAN PEMBAHASAN

a. Use Case Diagram

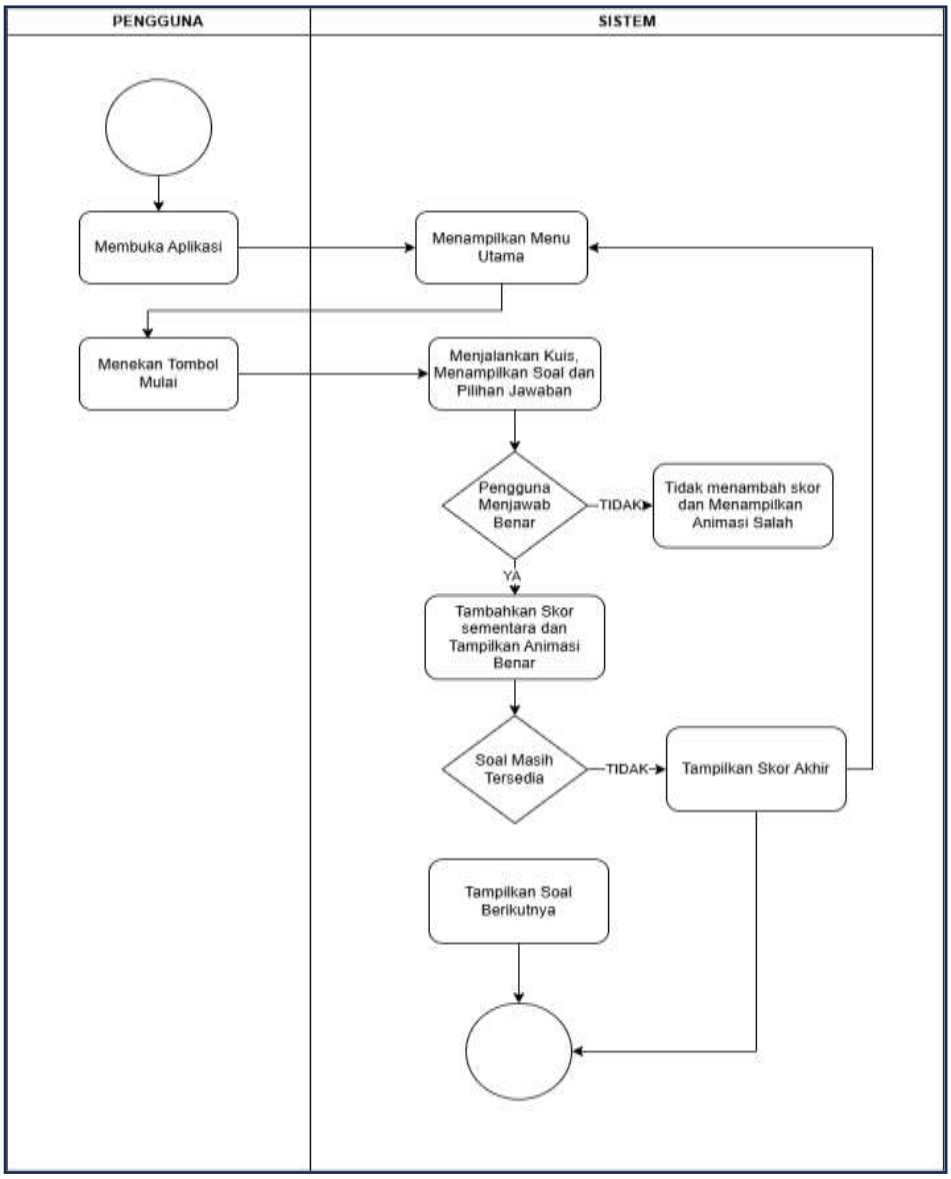
Use Case Diagram merupakan salah satu diagram dalam Unified Modeling Language (UML) yang berfungsi untuk memvisualisasikan hubungan antara aktor, baik pengguna maupun sistem lain, dengan sistem yang tengah dikembangkan. Menurut Fauzan (2021), diagram ini sering diterapkan dalam bidang pendidikan ilmu komputer dan rekayasa perangkat lunak karena kesederhanaannya, namun tetap efektif dalam merepresentasikan interaksi antara sistem dan aktor. Dibawah ini Use Case Diagram dari aplikas game kuis sederhana yang kami buat.



gambar 1. Use Case Diagram Game Kuis

b. Activity Diagram

Activity Diagram adalah salah satu diagram dalam Unified Modeling Language (UML) yang digunakan untuk memodelkan alur kerja atau aktivitas dalam suatu sistem. Diagram ini menggambarkan urutan eksekusi dari aktivitas yang dilakukan, baik dalam bentuk proses bisnis maupun interaksi dalam perangkat lunak (Kundu & Samanta, 2009). Dibawah ini adalah activity diagram dari aplikasi game kuis sederhana yang kami buat.



gambar 2. Activity Diagram Game Kuis

c. Tampilan Aplikasi

1. Menu Utama

gambar 3. *Tampilan Menu Utama*

Pada saat user membuka aplikasi, user akan langsung diarahkan ke menu utama aplikasi yang terdiri dari judul aplikasi, tombol play, dan tombol keluar. Tombol Play berfungsi untuk memulai kuis dan tombol Exit berfungsi untuk keluar dari aplikasi.

2. Tampilan Kuis

gambar 4. *Tampilan Kuis*

Pada tampilan kuis terdapat soal dan pilihan jawaban sebanyak 3 pilihan dibawahnya. Jika pemain menjawab benar maka skor akan bertambah

sebanyak 10 dan menampilkan animasi benar. Jika pemain menjawab salah maka skor tidak akan bertambah dan menampilkan animasi salah. Jika soal masih tersedia makanya sistem akan menampilkan soal berikutnya sebaliknya jika soal sudah habis maka akan menampilkan skor akhir.

gambar 5. *Tampilan Nilai Akhir*

d. Kodingan

Aplikasi ini di program menggunakan bahasa C *Sharp* dan menggunakan unity sebagai *game engine* nya. Berikut penjelasan kodingannya.

1. Library yang digunakan

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;
using TMPro;
```

- **System.Collections & System.Collections.Generic** → Digunakan untuk mendukung penggunaan array dan koleksi data.
- **UnityEngine** → Library utama Unity untuk mengakses komponen game.
- **UnityEngine.UI** → Digunakan untuk mengelola elemen antarmuka

pengguna seperti tombol dan teks.

- **TMPro (TextMeshPro)** → Digunakan untuk menangani teks dengan fitur tambahan dari TextMeshPro.

2. Deklarasi Kelas dan Struktur Data

```
public class QUIZ : MonoBehaviour
```

- **QUIZ** adalah kelas utama yang digunakan untuk menangani sistem kuis.
- **MonoBehaviour** adalah superclass dari semua skrip yang dapat digunakan dalam Unity.

3. Deklarasi Kelas Soal

```
[System.Serializable]
public class Soal
```

- **[System.Serializable]** → Membuat objek dari kelas ini bisa ditampilkan di Unity Inspector.
- **Soal** → Kelas yang digunakan untuk menyimpan data pertanyaan.

Atribut dalam kelas Soal:

```
[TextArea(3, 10)]
[Header("soal")]
public string soal;
```

- **soal** → Menyimpan teks pertanyaan dengan tampilan textarea di Unity.

```
[Header("pilihan")]
public string[] pilihan;
```

- **pilihan** → Array yang menyimpan opsi jawaban.

```
[Header("kunci jawaban")]
public bool[] jawaban;
```

- **jawaban** → Array boolean yang menyimpan kunci jawaban benar.

```
[Header("nilai")]
public int nilai;
```

- **nilai** → Skor yang diberikan untuk setiap pertanyaan.

4. Deklarasi Variabel dan Objek

```
public Soal[] soal;
int urutan_soal = -1, nilai = 0;
```

- **soal[]** → Array dari kelas **Soal** untuk menyimpan daftar pertanyaan.
- **urutan_soal** → Indeks pertanyaan yang sedang ditampilkan.
- **nilai** → Menyimpan skor pengguna.

```
public Button[] jawaban;
public Animator gameplay;
public TMP_Text text_soal, text_skor, text_nilaiAKhir;
public GameObject feedback_bnr, feedback_slh, hasil_akhir;
public AudioSource suara_bnr, suara_slh, backsound;
```

- **jawaban** → Array tombol untuk jawaban.
- **gameplay** → Animator untuk animasi tampilan kuis.
- **text_soal** → Teks pertanyaan yang akan ditampilkan.
- **text_skor & text_nilaiAkhir** → Menampilkan skor saat ini dan nilai akhir.
- **feedback_bnr & feedback_slh** → Efek visual untuk jawaban benar atau salah.
- **hasil_akhir** → Panel hasil akhir kuis.
- **suara_bnr & suara_slh** → Efek suara untuk jawaban benar dan salah.
- **backsound** → Musik latar selama kuis berlangsung.

5. Fungsi Menampilkan Soal

```
public void set_soal()
{
    urutan_soal++;
    if (urutan_soal < soal.Length)
    {
        gameplay.Play(0);
        text_soal.text = soal[urutan_soal].soal;
        for (int i = 0; i < jawaban.Length; i++)
        {
            jawaban[i].transform.GetChild(0).GetComponent<TMP_Text>().text = soal[urutan_soal].pilihan[i];
        }
    }
    else
    {
        gameplay.gameObject.SetActive(false);
        hasil_akhir.SetActive(true);
        text_nilaiAKhir.text = nilai.ToString();
        backsound.Stop();
    }
}
```

- Meningkatkan indeks pertanyaan.
- Menampilkan pertanyaan dan pilihan jawaban.
- Jika semua soal selesai, kuis berhenti dan tampilan hasil akhir muncul.

6. Fungsi Mengecek Jawaban

```
public void jawab(int indeks_pilihan)
{
    bool jbn_benar = false;
    for(int i = 0; i < jawaban.Length; i++)
    {
        if(indeks_pilihan == i && soal[urutan_soal].jawaban[i])
        {
            jbn_benar = true;
            i = jawaban.Length;
        }
    }
    if (jbn_benar)
    {
        feedback_bnr.SetActive(true);
        feedback_bnr.GetComponent<Animator>().Play(0);
        nilai += soal[urutan_soal].nilai;
        suara_bnr.Play(0);
    }
    else
    {
        feedback_slh.SetActive(true);
        feedback_slh.GetComponent<Animator>().Play(0);
        suara_slh.Play(0);
    }
}
```

- Mengecek apakah jawaban yang dipilih benar atau salah.
- Jika benar, menampilkan animasi dan menambah nilai.
- Jika salah, menampilkan animasi kesalahan.

7. Fungsi Memperbarui Skor

```
public void output()
{
    text_skor.text = nilai.ToString();
}
```

- Memperbarui tampilan skor pada UI.

8. Metode Start() dan Update()

```
void Start()
{
    set_soal();
    gameplay.GetComponent<Animator>().Play(0);
}
```

- Memulai kuis dengan menampilkan soal pertama.
- Memutar animasi awal.

```
void Update()
{
    output();
}
```

- Memastikan skor selalu diperbarui setiap frame.

9. Konsep OOP dalam kode ini

Kode ini menerapkan beberapa prinsip Object-Oriented Programming (OOP):

a. Encapsulation (Enkapsulasi)

Enkapsulasi berarti menyembunyikan detail implementasi suatu objek dan hanya mengekspos informasi yang diperlukan.

- Kelas **Soal** menyimpan data soal secara terstruktur.
- Variabel seperti nilai dan **urutan_soal** hanya dapat diakses dalam kelas ini.

b. Abstractio (Abstraksi)

Abstraksi memungkinkan kode untuk berfokus pada fitur penting tanpa

menampilkan detail implementasi.

- Fungsi **set_soal(), jawab()**, dan **output()** menyederhanakan logika program

c. Inheritance (Pewarisan)

```
public class QUIZ : MonoBehaviour
```

Kelas QUIZ mewarisi MonoBehaviour, sehingga bisa digunakan sebagai skrip Unity.

d. Polymorphism (Polimorfisme)

Polimorfisme memungkinkan metode yang sama digunakan dengan cara yang berbeda.

- Fungsi **jawab(int indeks_pilihan)** mengevaluasi semua kemungkinan jawaban tanpa mengetahui jumlah jawaban secara eksplisit

4. KESIMPULAN

Penelitian ini bertujuan untuk mengimplementasikan paradigma Pemrograman Berorientasi Objek (PBO/OOP) dalam pengembangan game kuis interaktif. Game ini dikembangkan menggunakan game engine Unity dan bahasa pemrograman C#, yang dipilih karena kemampuannya dalam mendukung pengembangan aplikasi berbasis OOP serta kemudahan dalam pengintegrasian berbagai komponen.

Aplikasi game kuis yang dikembangkan terdiri dari beberapa fitur utama, yaitu tampilan utama, tampilan kuis, dan skor akhir. Game ini telah diuji dan dapat berjalan dengan baik sesuai dengan tujuan yang dirancang.

Namun, saya menyadari bahwa penelitian ini masih memiliki beberapa keterbatasan yang perlu diperbaiki. Oleh karena itu, penelitian lebih lanjut diperlukan untuk mengoptimalkan penerapan konsep OOP dalam pengembangan game, meningkatkan efisiensi kode, serta memperluas fitur agar aplikasi lebih interaktif dan adaptif terhadap berbagai kebutuhan pengguna.

DAFTAR PUSTAKA

Nagineni, R.B. (2021). *A Research on Object-Oriented Programming and Its Concepts*. International Journal of Advanced Trends in Computer Science and Engineering.

Savinov, A.A. (2008). *Concepts and Concept-Oriented Programming*. Journal of Object Technology

Hussain, A., Shakeel, H. & Hussain, F. (2020). *Unity Game Development Engine: A Technical Survey*. University of Sindh Journal of Information and Communication Technology.

Nieminen, T. (2021). *Unity Game Engine in Visualization, Simulation, and Modelling*. BS Thesis, Core.

Prastowo, W.D., & Danianti, D. (2023). *Analisis Risiko pada Pengembangan Perangkat Lunak Menggunakan Metode Agile dan RAD (Rapid Application Development)*. Citizen: Jurnal Ilmiah.

Fauzan, R., Siahaan, D., & Rochimah, S. (2021). *A different approach on automated use case diagram semantic assessment*.

Kundu, D., & Samanta, D. (2009). *A novel approach to generate test cases from UML activity diagrams*. Journal of Object Technology