LAPORAN PROYEK AKHIR PEMOGRAMAN BERORIENTASI OBJEK APLIKASI VIRTUAL PET-KENMOCHI TOUYA

Mata Kuliah:

Pemograman Berorientasi Objek

Oleh

Muhammad Wahyu Firmansyah

24091397035

2024B

PROGRAM STUDI MANAJEMEN INFORMATIKA

FAKULTAS VOKASI

UNIVERSITAS NEGERI SURABAYA

2025

A. PENDAHULUAN

1. Latar Belakang

Proyek akhir ini terinspirasi dari konsep permainan klasik Tamagotchi, yang merupakan studi kasus ideal untuk penerapan Pemrograman Berorientasi Objek (PBO) dalam memodelkan interaksi antara karakter digital dan lingkungannya. Untuk memberikan identitas visual yang unik, penulis memilih karakter Kenmochi Touya (seorang Virtual YouTuber) sebagai tokoh utama, yang secara khusus diadaptasi ke dalam wujud kucing bergaya seni piksel (pixel art). Penyesuaian visual menjadi bentuk hewan ini dipilih agar selaras dengan tema Virtual Pet, sekaligus memberikan tampilan yang lebih ikonik dan menarik bagi pengguna.

Tantangan teknis utama dalam pengembangan aplikasi ini adalah menciptakan sistem interaksi yang dinamis dan persisten. Oleh karena itu, penulis mengimplementasikan logika State Machine untuk menangani animasi perpindahan ruangan (cutscene) yang halus, serta sistem penyimpanan data berbasis JSON agar progres permainan dapat disimpan. Melalui proyek ini, penulis bertujuan mendemonstrasikan kompetensi dalam menerapkan pilar-pilar OOP, khususnya Encapsulation dan Inheritance, menggunakan bahasa pemrograman Python dan pustaka Pygame.

2. Rumusan Masalah

Berdasarkan latar belakang tersebut, rumusan masalah yang diangkat dalam proyek ini adalah:

- Bagaimana merancang struktur kelas (Class) yang mampu merepresentasikan karakter kucing virtual dengan atribut dinamis (Lapar, Energi, Kebersihan, dan Kebahagiaan)?
- Bagaimana menerapkan konsep Inheritance dan Encapsulation untuk memanipulasi status karakter secara aman dan terstruktur?
- Bagaimana mengimplementasikan logika State Machine untuk menciptakan animasi perpindahan ruangan (cutscene) yang sinematik saat karakter beraktivitas?
- Bagaimana membangun sistem penyimpanan data (Save/Load) menggunakan format JSON agar progres permainan dapat disimpan dan dilanjutkan kembali?

3. Tujuan

Tujuan dari pengembangan aplikasi "Kenmochi Touya - Virtual Pet" ini adalah:

- **Tujuan Akademis:** Memenuhi tugas Proyek Akhir mata kuliah Pemrograman Berorientasi Objek (PBO) dengan menerapkan prinsip-prinsip utama OOP dalam kode program.

- **Tujuan Teknis:** Menghasilkan aplikasi simulasi yang stabil, memiliki fitur animasi transisi yang mulus, serta kemampuan manajemen data yang persisten.
- **Tujuan Fungsional:** Menyediakan sarana hiburan interaktif yang unik bagi pengguna melalui simulasi perawatan hewan peliharaan virtual dengan visualisasi karakter yang menarik.

B. ANALISIS DAN PERANCANGAN SISTEM

1. Deskripsi Sistem

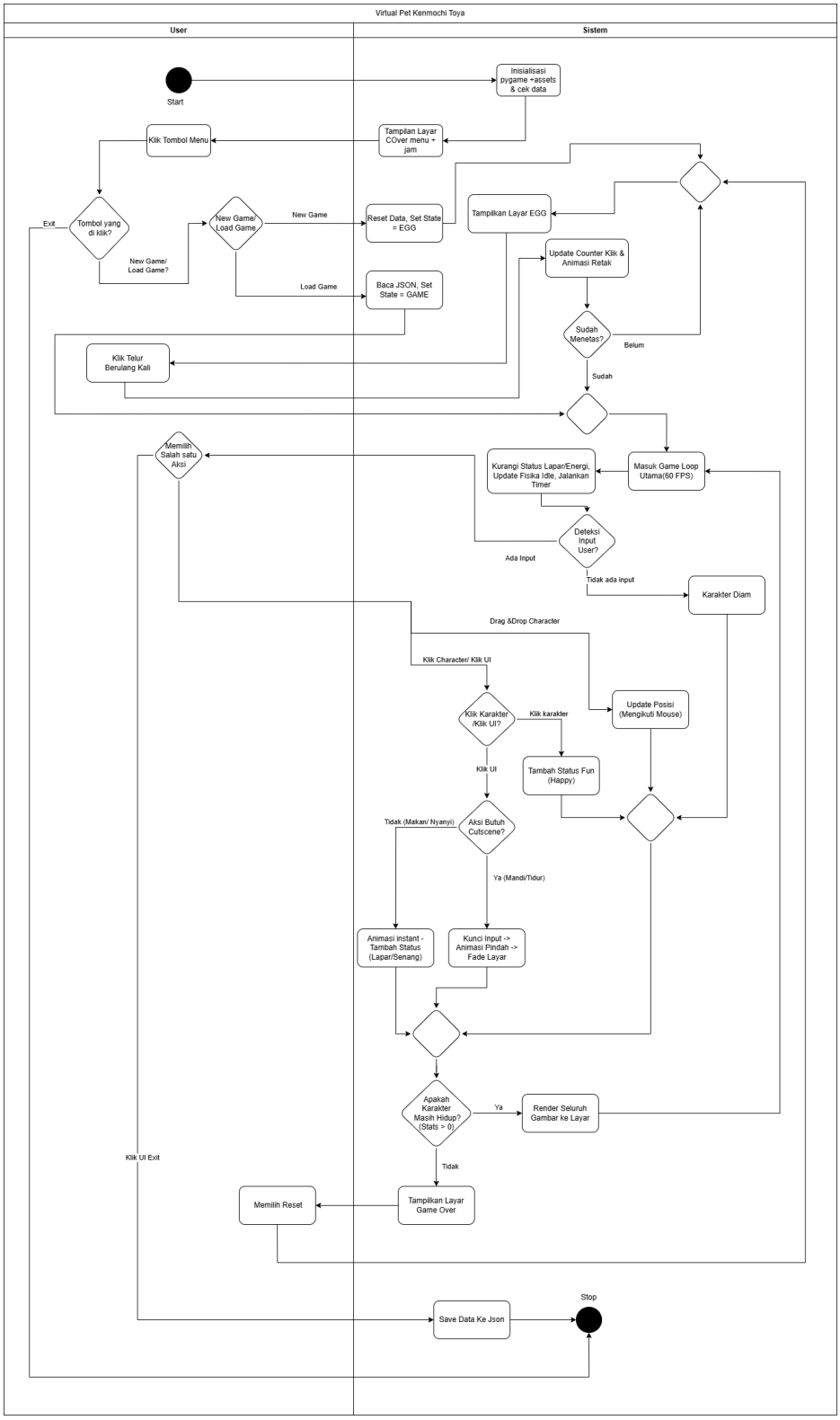
Aplikasi "Kenmochi Touya - Virtual Pet" adalah simulasi interaktif berbasis desktop yang menggunakan arsitektur Game Loop. Sistem berjalan secara real-time dengan kecepatan 60 FPS (Frame Per Second). Secara garis besar, sistem memiliki tiga state (kondisi) utama yang dikelola oleh Class Game:

- **cover**: Menampilkan menu utama dan waktu sistem.
- **egg**: Fase interaktif awal (menetaskan telur).
- **game**: Fase simulasi utama di mana logika perawatan karakter berjalan.

2. Alur Logika Program (Flowchart)

Alur kerja aplikasi dirancang mengikuti logika Game Loop yang berjalan terus-menerus hingga aplikasi ditutup. Berikut adalah deskripsi alur logika program dari awal hingga akhir:

1) Inisialisasi: Program memulai pustaka Pygame, memuat aset gambar/suara, dan mengecek ketersediaan file penyimpanan (savegame.json).
2) Menu Utama (Cover): Program menampilkan menu awal. Pengguna memilih antara memulai permainan baru (New Game) atau memuat progres lama (Load Game).
3) Fase Permainan:
    - **Jika New Game**: Masuk ke Fase Telur. Pengguna harus menetaskan telur terlebih dahulu sebelum masuk ke permainan utama.
    - **Jika Load Game**: Membaca data status terakhir dari JSON dan langsung masuk ke Fase Gameplay.
4) Siklus Utama (Main Loop):
    - **Input Handling**: Sistem mendeteksi interaksi mouse (klik tombol menu, usap kepala, atau drag & drop).
    - **Update Logic**: Mengurangi nilai status (Lapar, Energi, Kebersihan) secara berkala dan menjalankan logika animasi (cutscene).
    - **Cek Kondisi**: Memeriksa apakah status karakter mencapai 0 (Mati). Jika mati, tampilkan layar Game Over.
    - **Rendering**: Menggambar seluruh elemen grafis ke layar.
5) Terminasi: Saat pengguna menutup aplikasi, sistem secara otomatis menyimpan data status terkini ke file JSON (Auto-save) sebelum program berhenti.



Gambar 1 Diagram Aktivitas Virtual Pet Kenmochi Touya

3. Penerapan Konsep PBO (Object-Oriented Programming)

Aplikasi ini dibangun menggunakan paradigma PBO untuk memastikan kode yang modular dan terstruktur. Berikut adalah analisis penerapannya berdasarkan kode program:

a. Class dan Object

Program mendefinisikan kelas sebagai cetak biru (blueprint) dan menciptakan objek saat program dijalankan (runtime).

- **Class Game:** Bertindak sebagai Controller utama yang mengelola aset, input pengguna, dan perpindahan scene.
- **Class Kenmochi:** Merepresentasikan karakter utama beserta seluruh logikanya.
- **Class Egg:** Merepresentasikan objek fase awal permainan.
- **Class Button:** Objek antarmuka pengguna (UI) yang dapat digunakan kembali (reusable) untuk berbagai menu.

b. Inheritance (Pewarisan)

Pewarisan digunakan agar objek karakter dapat memanfaatkan fungsionalitas dasar dari pustaka Pygame tanpa perlu menulis ulang kode manajemen grafis. Kelas Kenmochi mewarisi kelas pygame.sprite.Sprite.

```python
# Class Kenmochi mewarisi (extends) sifat dari pygame.sprite.Sprite
class Kenmochi(pygame.sprite.Sprite):
    def __init__(self):
        super().__init__()  # Memanggil konstruktor kelas induk
        self.name = "Kenmochi"
        self.rect = pygame.Rect(0,0,0,0) # Inisialisasi properti Rect dari Sprite

        # ... (Kode inisialisasi stats lainnya)

        self.load_assets() # Memanggil method untuk memuat gambar
```

Dengan mewarisi Sprite, objek Kenmochi memiliki atribut standar self.rect yang sangat berguna untuk menangani posisi koordinat dan deteksi interaksi (collision). Penulis kemudian memperluas fungsionalitas kelas ini dengan menambahkan metode khusus seperti update_physics() untuk gravitasi dan update_cutscene() untuk animasi.

c. Encapsulation (Pembungkusan)

Konsep ini diterapkan untuk melindungi data vital karakter agar tidak dimanipulasi secara tidak valid. Atribut status disimpan dalam dictionary self.stats dan logika perubahannya dibungkus dalam metode update_stats().

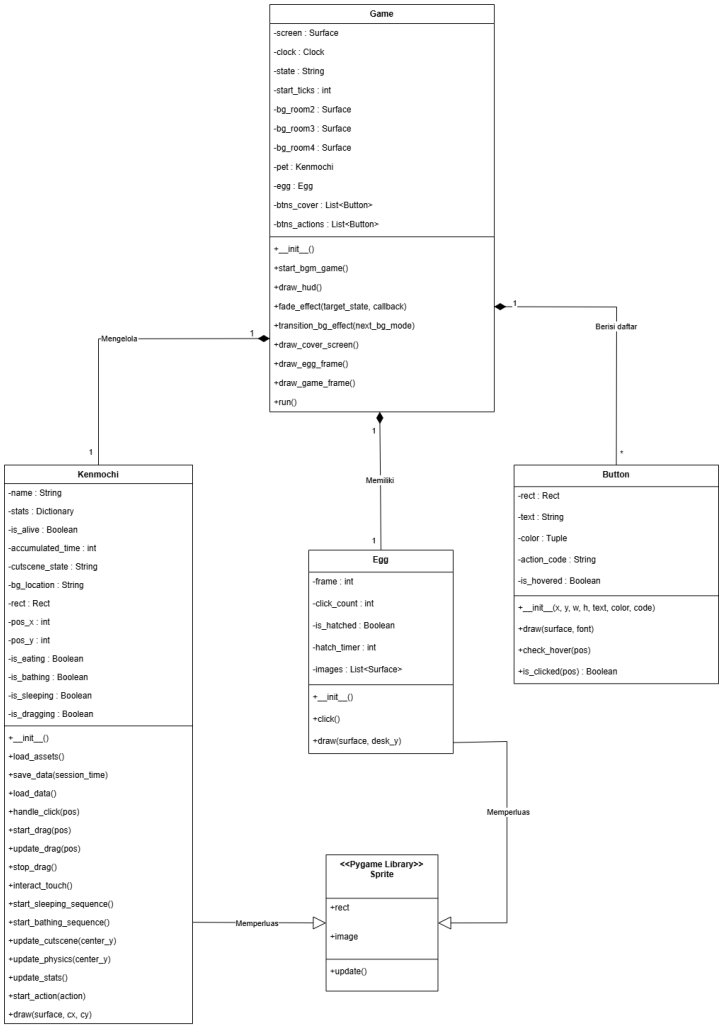
```python
def update_stats(self):
        # Logika pengurangan status terjadi secara internal (Private Logic)
        decay = 0.0185
        if not self.is_eating:
            self.stats["hunger"] -= 0.02

        # Validasi nilai agar tetap dalam rentang aman (0-100)
        for k in self.stats:
            self.stats[k] = max(0, min(100, self.stats[k]))
```

Kode di atas menunjukkan bagaimana sistem "menjaga" nilai status agar tidak pernah bernilai negatif (di bawah 0) atau berlebih (di atas 100) menggunakan fungsi min dan max, yang merupakan bentuk proteksi data.

4. Rancangan Desain (Class Diagram)

Berdasarkan struktur kode, hubungan antar-objek dalam sistem adalah sebagai berikut:

- Game (Main Class) memiliki hubungan Komposisi (Composition) dengan objek Kenmochi dan Egg. Artinya, objek karakter dan telur dibuat dan hidup di dalam siklus hidup objek Game.
- Game memiliki hubungan Agregasi (Aggregation) dengan objek Button, di mana Game menampung daftar tombol (self.btns_actions) untuk keperluan antarmuka.
- Kenmochi dan Egg adalah Turunan (Inheritance) dari kelas pygame.sprite.Sprite.



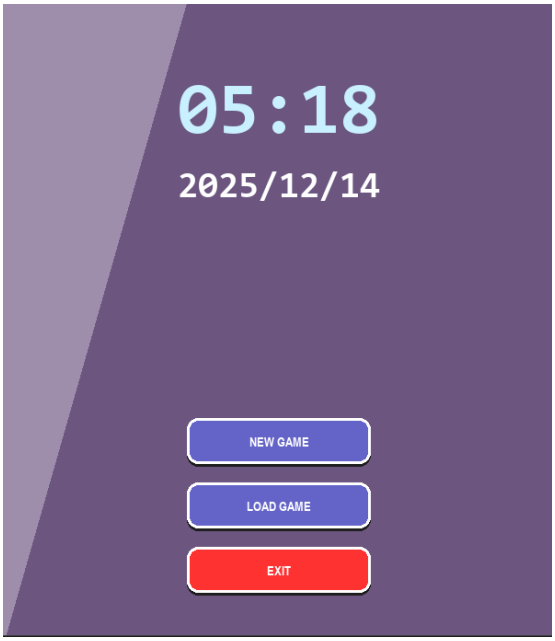
Gambar 2 Diagram kelas (Class Diagram) aplikasi Kenmochi Touya.

C. IMPLEMENTASI SISTEM

Bagian ini membahas implementasi antarmuka (interface) dan logika program yang telah dibangun menggunakan bahasa pemrograman Python dan pustaka Pygame.

1. Tampilan Menu Utama (Cover Screen)

Saat aplikasi dijalankan, sistem pertama kali memuat aset dan memeriksa ketersediaan data penyimpanan. Pengguna disajikan menu utama yang berisi opsi "New Game", "Load Game", dan "Exit".



Gambar 3 Tampilan menu utama aplikasi.

Logika menu utama menangani interaksi tombol untuk memulai permainan baru atau memuat data lama. (Snippet diambil dari method run dan start_new pada Class Game):

```
# Logika tombol Menu Utama (Class Game)
if btn.is_clicked(mouse_pos):
    if btn.action_code == "NEW":
        # Reset data dan masuk fase telur
        if os.path.exists("savegame.json"): os.remove("savegame.json")
        self.pet = Kenmochi(); self.egg = Egg()
        self.fade_effect("EGG", start_new) # Transisi ke Fase Telur

    elif btn.action_code == "LOAD":
        # Muat data dan masuk game
        self.load_data()
        self.fade_effect("GAME", start_load) # Transisi ke Fase Game
```

2. Fase Menetaskan Telur (Egg Stage)

Jika pengguna memilih new game, karakter tidak langsung muncul. Pengguna harus berinteraksi dengan objek telur dengan cara mengkliknya berulang kali hingga menetas.

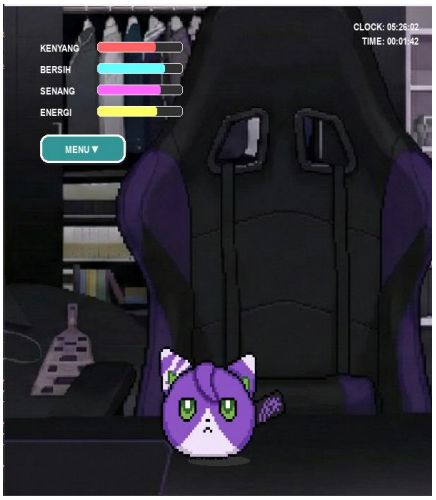
Gambar 4 5 6 Fase interaktif menetaskan telur

Sistem menghitung jumlah klik dan mengubah frame gambar telur secara bertahap.

```python
# Logika penetasan telur (Class Egg)
def click(self):
    if self.is_hatched: return
    self.click_count += 1

    # Perubahan frame berdasarkan jumlah klik
    if self.click_count >= 3 and self.click_count < 6:
        self.frame = 1  # Telur mulai retak
    elif self.click_count >= 6:
        self.frame = 2  # Telur pecah
        self.is_hatched = True # Status menetas aktif
```

3. Antarmuka Gameplay Utama

Ini adalah tampilan utama simulasi. Di sini pengguna dapat melihat karakter "Kenmochi Touya" melakukan aktivitas animasi Idle dan melihat status bar (Lapar, Energi, Kebersihan, Kesenangan) di pojok kiri atas.



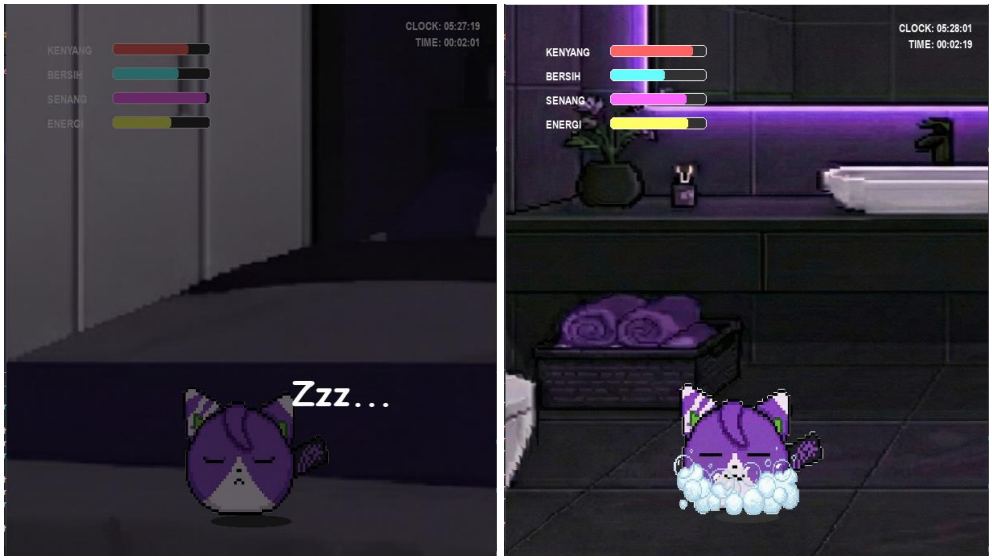
Gambar 7 Tampilan utama permainan (Gameplay).

Status karakter diperbarui secara real-time di setiap frame. Logika pengurangan status (decay) terjadi jika karakter tidak sedang tidur atau makan. (Snippet diambil dari Class Kenmochi method update_stats):

```
1
2  # Update status (Class Kenmochi)
3  def update_stats(self):
4      if not self.is_alive: return
5      decay = 0.0185
6
7      # Pengurangan status otomatis
8      if not self.is_eating: self.stats["hunger"] -= 0.02
9      if not self.is_sleeping: self.stats["energy"] -= decay
10     if not self.is_singing: self.stats["fun"] -= 0.015
11     if not self.is_bathing: self.stats["hygiene"] -= 0.01
```

4. Fitur Interaksi dan Animasi (Cutscene)

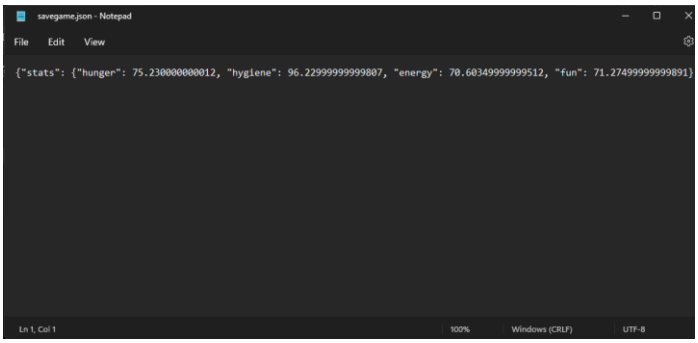
Salah satu fitur unggulan sistem ini adalah adanya cutscene saat melakukan aksi tertentu, seperti mandi atau tidur. Saat tombol ditekan, input pengguna dikunci sementara hingga animasi selesai.



Gambar 8 dan 9 Contoh cutscene saat karakter sedang tidur dan mandi.

5. Implementasi Penyimpanan Data (JSON)

Sistem menggunakan format JSON (JavaScript Object Notation) untuk menyimpan progres permainan secara persisten. Data disimpan otomatis saat aplikasi ditutup.



D. PENUTUP

1. Kesimpulan

Berdasarkan tahapan analisis, perancangan, hingga implementasi yang telah dilakukan dalam pembuatan aplikasi "Kenmochi Touya - Virtual Pet", dapat ditarik beberapa kesimpulan sebagai berikut:

- **Keberhasilan Implementasi PBO**: Aplikasi berhasil dibangun dengan menerapkan prinsip Object-Oriented Programming (PBO) secara utuh. Penggunaan Class dan Object membuat struktur kode lebih terorganisir, sementara konsep Inheritance (pewarisan sifat pygame.sprite.Sprite) dan Encapsulation (proteksi variabel status) telah meningkatkan efisiensi dan keamanan data program.
- **Fungsionalitas Sistem**: Logika Game Loop berjalan stabil pada kecepatan 60 FPS, mampu menangani input pengguna (mouse events), memperbarui logika permainan secara real-time, dan merender animasi tanpa lag.
- **Penyimpanan Data**: Fitur persistensi data menggunakan format JSON berhasil diimplementasikan, memungkinkan pengguna untuk menyimpan dan melanjutkan progres permainan (status Lapar, Energi, dll.) meskipun aplikasi telah ditutup.

2. Saran

Meskipun aplikasi ini telah berjalan sesuai rancangan dasar, masih terdapat beberapa aspek yang dapat dikembangkan untuk versi selanjutnya:

- **Penambahan Fitur Mini-Game**: Agar permainan tidak monoton, dapat ditambahkan fitur permainan kecil (mini-games) di dalam aplikasi untuk menambah status kebahagiaan (Fun) secara lebih interaktif.
- **Peningkatan Aset Audio**: Saat ini sistem hanya memiliki musik latar (BGM). Penambahan efek suara (Sound Effects/SFX) pada setiap tombol atau aksi (seperti suara makan atau tidur) akan membuat pengalaman bermain lebih hidup.
- **Kompleksitas Animasi**: Menambahkan variasi animasi yang lebih halus (lebih banyak frame) dan efek visual partikel saat berinteraksi dengan karakter.
- **Porting ke Mobile**: Mengingat sifat permainan Virtual Pet yang kasual, pengembangan ke platform mobile (Android/iOS) menggunakan kerangka kerja seperti Kivy atau integrasi Pygame ke Android akan membuat aplikasi lebih mudah diakses.

E. DAFTAR PUSTAKA

Python Software Foundation. (2024). *Python 3.12 Documentation*. Diakses dari https://docs.python.org/3/

Shinners, P., et al. (2024). *Pygame Documentation*. Diakses dari https://www.pygame.org/docs/

Rumbaugh, J., Jacobson, I., & Booch, G. (2004). *The Unified Modeling Language Reference Manual (2nd Edition)*. Addison-Wesley Professional.

Gamma, E., Helm, R., Johnson, R., & Vlissides, J. (1994). *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley.

Nugroho, A. (2010). *Rekayasa Perangkat Lunak Berorientasi Objek dengan Metode USDP*. Yogyakarta: Andi Offset.