LAPORAN TUGAS AKHIR
GRAFIKA KOMPUTER DAN MULTIMEDIA

Pembuatan Aplikasi Advanced Drawing App
Menggunakan Python Tkinter

Disusun oleh

Nama : **Windy Claudia Napitupulu**
NIM : **123220029**
Kelas : **IF – A**

PROGRAM STUDI INFOMRATIKA
JURUSAN INFORMATIKA
FAKULTAS TEKNIK INDUSTRI
UNIVERSITAS PEMBANGUNAN NASIONAL "VETERAN"
YOGYAKARTA
2025

HALAMAN PENGESAHAN
LAPORAN TUGAS AKHIR
GRAFIKA KOMPUTER DAN MULTIMEDIA

Pembuatan Aplikasi Advanced Drawing App
Menggunakan Python Tkinter

Disusun Oleh :

Windy Claudia Napitupulu 123220029

Telah diperiksa dan disetujui oleh
Dosen Pengampu Mata Kuliah Grafika Komputer dan Multimedia
Pada tanggal : 17 Juni 2025

.

Menyetujui.

Dosen Pengampu Mata Kuliah
Grafika Komputer dan Multimedia

Frans Richard Kodong, S.T., M.Kom., Ph.D
NIDN. 0523026201

KATA PENGANTAR

Puji syukur penulis panjatkan ke hadirat Tuhan Yang Maha Esa atas segala rahmat dan karunia-Nya sehingga penulis dapat menyelesaikan proyek dan laporan dengan judul "Pembuatan Aplikasi Advanced Drawing App Menggunakan Python Tkinter". Laporan ini disusun sebagai bentuk dokumentasi dari proses perancangan, implementasi, dan evaluasi aplikasi grafika komputer berbasis antarmuka grafis (GUI).

Penulis menyampaikan terima kasih kepada Dosen pengampu mata kuliah Grafika Komputer dan Multimedia yaitu bapak Frans Richard Kodong, S.T., M.Kom., Ph.D yang telah memberikan bimbingan serta kesempatan untuk mengembangkan proyek ini. Ucapan terima kasih juga ditujukan kepada rekan-rekan yang telah memberikan dukungan teknis maupun moral selama proses pengerjaan proyek.

Yogyakarta, 17 Juni 2025

Windy Claudia Napitupulu

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR LAMPIRAN

BAB I
PENDAHULUAN

1.1 Latar Belakang

Dalam dunia teknologi modern, kemampuan untuk berinteraksi dengan sistem grafis menjadi sangat penting, baik untuk keperluan pendidikan, desain, maupun hiburan. Penerapan grafika komputer dalam bentuk aplikasi menggambar memberikan kesempatan bagi pengguna untuk mengekspresikan kreativitas secara digital. Oleh karena itu, dengan memanfaatkan bahasa pemrograman Python yang mudah dipelajari dan pustaka GUI Tkinter yang tersedia secara default, proyek ini bertujuan untuk mengembangkan aplikasi menggambar interaktif bernama "Advanced Drawing App".

Aplikasi ini menyediakan berbagai alat gambar seperti pensil, crayon, penghapus, serta bentuk-bentuk geometris seperti garis, persegi, dan oval. Selain itu, aplikasi ini mendukung fitur tambahan seperti pemilihan warna, pengaturan ketebalan kuas, undo dan redo, serta penyimpanan hasil gambar dalam format PNG. Dengan fitur-fitur tersebut, aplikasi ini dapat dijadikan media latihan serta memperkuat pemahaman konsep event-driven programming, manajemen objek visual, dan manipulasi canvas.

1.2 Rumusan Masalah

Bagaimana merancang dan membangun aplikasi menggambar berbasis Python yang mendukung berbagai fitur interaktif seperti pilihan warna, alat gambar beragam, undo/redo, serta penyimpanan hasil gambar dalam format file?

1.3 Tujuan

Membuat aplikasi gambar digital berbasis GUI dengan fitur:

1. Pilihan alat: pensil, crayon, penghapus, garis, persegi, oval
2. Fitur warna dan ketebalan kuas
3. Undo/redo dan clear canvas
4. Input teks pada kanvas
5. Simpan gambar sebagai file PNG

1.4 Manfaat

Manfaat yang diharapkan dari tugas praktik ini meliputi:

1. Memberikan pemahaman mendalam tentang pemrograman GUI menggunakan Tkinter
2. Meningkatkan keterampilan pengolahan gambar digital
3. Menjadi proyek edukatif untuk belajar tentang manajemen event dan objek visual
4. Sebagai fondasi awal untuk membangun aplikasi grafika komputer yang lebih kompleks di masa depa

BAB II
TINJAUAN LITERATUR

2.1 Grafika Komputer

Grafika komputer merupakan cabang dari ilmu komputer yang berkaitan dengan pembuatan, manipulasi, dan penyajian data visual dalam bentuk gambar dua dimensi maupun tiga dimensi. Grafika komputer sangat berperan dalam berbagai bidang, seperti pendidikan, hiburan, desain grafis, dan simulasi ilmiah. Dalam konteks aplikasi menggambar, grafika komputer memungkinkan pengguna untuk membuat dan memodifikasi elemen visual secara interaktif melalui berbagai alat dan teknik rendering.

Penerapan dasar grafika komputer dalam aplikasi menggambar mencakup konsep koordinat, penggambaran garis dan kurva, pengolahan warna, serta manajemen objek. Penggunaan canvas (kanvas digital) adalah media utama dalam penggambaran objek yang dapat dikontrol oleh pengguna melalui perangkat input seperti mouse.

2.2 Tkinter

Tkinter adalah pustaka antarmuka grafis (GUI) standar yang disertakan dalam distribusi Python. Tkinter menyediakan berbagai komponen seperti tombol (Button), label (Label), kanvas (Canvas), slider (Scale), menu, dan dialog yang memungkinkan pengembang untuk membuat aplikasi dengan tampilan antarmuka pengguna yang interaktif dan ramah pengguna.

Dalam aplikasi ini, Tkinter digunakan untuk membangun seluruh antarmuka, termasuk toolbar untuk pemilihan alat, slider ketebalan kuas, serta kanvas utama untuk menggambar. Keunggulan Tkinter adalah kemudahan implementasi dan integrasi langsung dengan Python tanpa memerlukan instalasi tambahan.

2.3 Python Imaging Library (Pillow)

Pillow adalah pengembangan dari Python Imaging Library (PIL) yang merupakan pustaka untuk manipulasi gambar. Library ini menyediakan fungsi untuk membuka, mengedit, dan menyimpan berbagai format gambar seperti PNG, JPEG, BMP, dan lainnya. Salah satu fitur penting Pillow adalah kemampuan untuk mengambil cuplikan layar atau area tertentu dari layar menggunakan ImageGrab.

Pada proyek ini, Pillow digunakan untuk menyimpan hasil gambar dari kanvas dalam format file PNG. Proses ini dilakukan dengan mengambil koordinat layar dari objek kanvas, lalu menyimpannya sebagai file gambar dengan menggunakan metode ImageGrab.grab().crop().

2.4 Drawing Tool dan Antarmuka Pengguna

Drawing tool adalah perangkat lunak atau bagian dari aplikasi yang menyediakan kemampuan untuk menggambar objek secara langsung di atas area kerja digital. Umumnya drawing tool mencakup alat seperti pensil, kuas, penghapus, bentuk-bentuk geometri (rectangle, circle), dan teks. Fitur tambahan seperti pengaturan warna, ukuran, serta undo dan redo juga menjadi bagian penting dari interaksi pengguna.

Antarmuka pengguna (User Interface/UI) yang baik dalam aplikasi drawing harus intuitif dan mudah diakses oleh pengguna. Dalam aplikasi ini, UI dirancang dengan layout vertikal yang menempatkan semua tool pada toolbar bagian atas, sementara area menggambar tersedia secara penuh di bagian tengah. Interaksi utama dikendalikan dengan event mouse seperti klik, drag, dan release.

Dengan menggabungkan konsep-konsep tersebut, pengguna dapat merasakan pengalaman menggambar yang responsif, fleksibel, dan menyenangkan, meskipun menggunakan teknologi yang tergolong ringan dan sederhana seperti Tkinter.

BAB III
METODOLOGI

3.1 Bahasa dan Tools yang Digunakan

Aplikasi *Advanced Drawing App* dikembangkan menggunakan bahasa pemrograman Python versi 3 karena kelebihan sintaks yang mudah dibaca, komunitas besar, serta dukungan pustaka GUI bawaan seperti Tkinter. Selain itu, aplikasi ini juga menggunakan library tambahan yaitu Pillow untuk mendukung manipulasi gambar digital.

Berikut adalah daftar tools yang digunakan:

- Python 3.x: Bahasa pemrograman utama.
- Tkinter: GUI toolkit bawaan Python.
- Pillow (PIL): Untuk menangkap dan menyimpan hasil gambar dari canvas.
- Visual Studio Code: Sebagai IDE (Integrated Development Environment) dalam proses pengembangan.
- Windows 10: Sistem operasi yang digunakan saat pengembangan dan pengujian.

3.2 Desain Antarmuka dan Toolbar

Desain antarmuka aplikasi disusun dengan mempertimbangkan kemudahan penggunaan oleh pengguna. Komponen utama dalam antarmuka meliputi:

- **Toolbar**: Berisi tombol-tombol untuk memilih alat seperti pensil, crayon, penghapus, garis, persegi, oval, teks, serta tombol tambahan untuk pemilihan warna, undo, redo, clear, dan simpan gambar.
- **Slider**: Berfungsi untuk mengatur ketebalan kuas gambar, mulai dari ukuran 1 hingga 20 piksel.
- **Canvas**: Area putih berukuran 800x600 piksel yang digunakan pengguna untuk menggambar dengan alat-alat yang tersedia.

Toolbar menggunakan ikon dari folder lokal icons, dan fallback ke teks jika gambar tidak ditemukan.

3.3 Implementasi Fitur

Setiap fitur dalam aplikasi diimplementasikan berdasarkan event dan interaksi pengguna. Beberapa fitur utama antara lain:

- Pensil dan Crayon: Menggambar garis bebas. Crayon memiliki tekstur kasar yang dihasilkan dengan garis-garis acak.
- Penghapus: Menggambar dengan warna putih untuk menyimulasikan penghapusan.
- Garis, Persegi, dan Oval: Dibuat menggunakan klik-drag mouse. Objek bersifat sementara selama drag dan permanen saat dilepas.
- Teks: Menggunakan simpledialog untuk input string teks yang ditempatkan di canvas.
- Pemilihan Warna: Menggunakan colorchooser untuk memilih warna kuas.
- Undo dan Redo: Menggunakan dua stack untuk menyimpan ID objek yang dibuat.
- Penyimpanan Gambar: Menggunakan ImageGrab untuk menyimpan canvas ke file PNG.

3.4 Struktur Program dan Alur Eksekusi

Struktur program mengikuti pendekatan berorientasi objek, dengan satu kelas utama DrawingApp. Komponen penting meliputi:

- __init__(): Inisialisasi komponen, state aplikasi, dan pengikatan event.
- setup_ui(): Menyusun elemen-elemen GUI termasuk toolbar dan canvas.
- select_tool(tool): Mengatur tool aktif.
- start_draw(event), draw(event), dan reset(event): Mengatur perilaku menggambar dari awal klik hingga pelepasan mouse.
- save_canvas(): Menyimpan isi canvas sebagai file gambar.

Alur eksekusi dikendalikan oleh event mouse dan pengelolaan objek pada canvas secara real-time.

BAB IV
HASIL DAN PEMBAHASAN

4.1 Hasil

Setelah proses perancangan dan implementasi selesai dilakukan, diperoleh sebuah aplikasi dengan antarmuka grafis yang berfungsi sebagai media menggambar digital sederhana. Aplikasi ini diberi nama *Advanced Drawing App* dan telah diuji dalam sistem operasi Windows dengan hasil sebagai berikut:

4.1.1 Tampilan Antarmuka

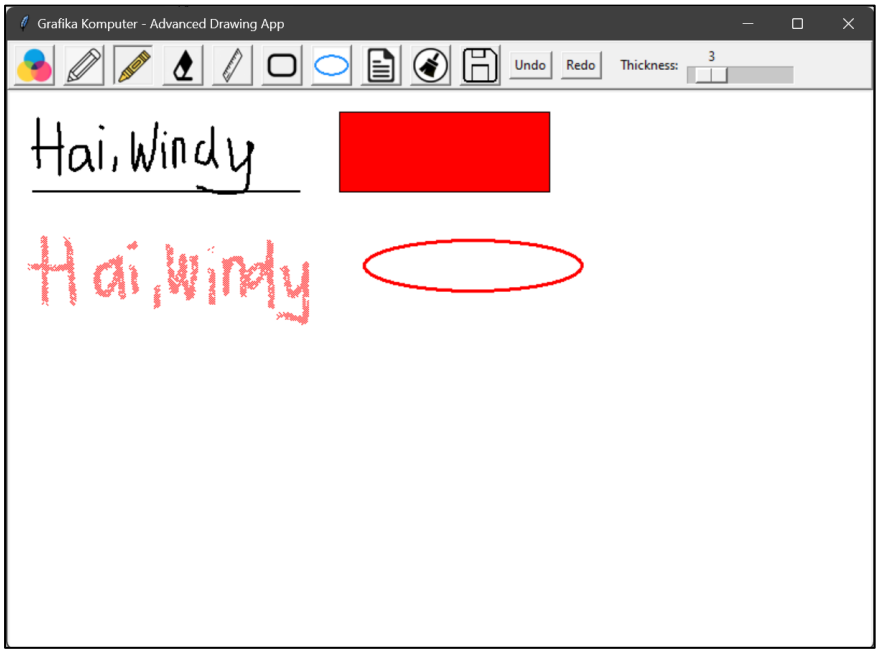
Aplikasi menampilkan jendela utama dengan:

- Toolbar di bagian atas berisi ikon alat gambar,
- Slider untuk pengaturan ketebalan kuas,
- Canvas sebagai area utama menggambar.

4.1.2 Fitur-Fitur yang Berhasil Diimplementasikan

- Pemilihan Alat: Pengguna dapat memilih alat seperti pensil, crayon, penghapus, garis, persegi panjang, oval, dan teks.
- Pengaturan Warna: Tersedia pemilih warna (color chooser) untuk menentukan warna kuas.
- Pengaturan Ketebalan Kuas: Pengguna dapat menyesuaikan ketebalan alat gambar melalui slider horizontal.
- Undo/Redo: Pengguna dapat membatalkan atau mengulangi tindakan menggambar.
- Clear Canvas: Seluruh isi canvas dapat dihapus untuk memulai gambar baru.
- Simpan Gambar: Gambar dapat disimpan dalam format PNG menggunakan library Pillow.

4.1.3 Dokumentasi Visual



Gambar 1. Tampilan Aplikasi

Tampilan antarmuka aplikasi Advanced Drawing App yang menunjukkan berbagai fitur utama seperti teks, garis bawah, bentuk persegi dengan isian warna merah, teks dengan efek crayon berwarna merah, dan bentuk oval. Toolbar di bagian atas menyediakan pilihan alat gambar, pengatur ketebalan kuas, serta tombol undo dan redo.

4.1.4 Source Code

```
import tkinter as tk
from tkinter import colorchooser, filedialog, simpledialog
from PIL import ImageTk, Image, ImageGrab
import os
import random
import time

class DrawingApp:
    def __init__(self, root):
        self.root = root
        self.root.title("Grafika Komputer - Advanced Drawing App")

        self.brush_color = "black"
        self.brush_size = 3
```

```
        self.tool = "pencil"

        self.last_x, self.last_y = None, None
        self.temp_shape = None
        self.start_time = 0

        self.icon_images = {}
        self.tool_buttons = {}
        self.tools = ["color", "pencil", "crayon", "eraser", "line",
"rect", "oval", "text", "clear", "save", "undo", "redo"]

        self.undo_stack = []
        self.redo_stack = []

        self.setup_ui()

    def setup_ui(self):
        self.toolbar = tk.Frame(self.root, bd=2, relief=tk.RAISED)
        self.toolbar.pack(side=tk.TOP, fill=tk.X)

        for tool in self.tools:
            img_path = os.path.join("icons", f"{tool}.png")
            if not os.path.exists(img_path):
                img_path = os.path.join("icons", f"{tool}.jpg")
            if not os.path.exists(img_path):
                img_path = os.path.join("icons", f"{tool}.jpeg")
            if os.path.exists(img_path):
                img = Image.open(img_path).resize((32, 32))
                self.icon_images[tool] = ImageTk.PhotoImage(img)
                btn           =           tk.Button(self.toolbar,
image=self.icon_images[tool],
                                  command=lambda             t=tool:
self.select_tool(t))
                btn.pack(side=tk.LEFT, padx=4, pady=2)
                self.tool_buttons[tool] = btn
            else:
                btn           =           tk.Button(self.toolbar,
text=tool.capitalize(), command=lambda t=tool: self.select_tool(t))
                btn.pack(side=tk.LEFT, padx=4, pady=2)
                self.tool_buttons[tool] = btn
```

```
        tk.Label(self.toolbar,
text="Thickness:").pack(side=tk.LEFT, padx=(10, 2))
        self.slider   =   tk.Scale(self.toolbar,   from_=1,   to=20,
orient=tk.HORIZONTAL, command=self.change_thickness)
        self.slider.set(self.brush_size)
        self.slider.pack(side=tk.LEFT)

        self.canvas  =  tk.Canvas(self.root,  bg="white",  width=800,
height=600)
        self.canvas.pack(fill=tk.BOTH, expand=True)

        self.canvas.bind("<ButtonPress-1>", self.start_draw)
        self.canvas.bind("<B1-Motion>", self.draw)
        self.canvas.bind("<ButtonRelease-1>", self.reset)

    def select_tool(self, tool):
        if tool == "color":
            color = colorchooser.askcolor()[1]
            if color:
                self.brush_color = color
        elif tool == "clear":
            self.canvas.delete("all")
            self.undo_stack.clear()
        elif tool == "save":
            self.save_canvas()
        elif tool == "undo":
            if self.undo_stack:
                item = self.undo_stack.pop()
                self.canvas.delete(item)
                self.redo_stack.append(item)
        elif tool == "redo":
            if self.redo_stack:
                item = self.redo_stack.pop()
                self.canvas.itemconfigure(item, state='normal')
                self.undo_stack.append(item)
        else:
            self.tool = tool
            for t, btn in self.tool_buttons.items():
```

```
                btn.config(relief=tk.SUNKEN  if  t  ==  tool  else
tk.RAISED)

    def change_thickness(self, val):
        self.brush_size = int(val)

    def start_draw(self, event):
        self.last_x, self.last_y = event.x, event.y
        self.start_time = time.time()
        self.temp_shape = None

    def draw(self, event):
        if self.tool == "pencil":
            item       =       self.canvas.create_line(self.last_x,
self.last_y, event.x, event.y,
                                            fill=self.brush_color,
width=self.brush_size,
                                            capstyle=tk.ROUND,
smooth=True)
            self.undo_stack.append(item)
            self.last_x, self.last_y = event.x, event.y

        elif self.tool == "crayon":
            for _ in range(3):
                offset_x = random.randint(-2, 2)
                offset_y = random.randint(-2, 2)
                item = self.canvas.create_line(
                    self.last_x + offset_x, self.last_y + offset_y,
                    event.x + offset_x, event.y + offset_y,
                    fill=self.brush_color,             width=max(1,
self.brush_size - 1),
                    capstyle=tk.ROUND,                 smooth=True,
stipple="gray50"
                )
                self.undo_stack.append(item)
            self.last_x, self.last_y = event.x, event.y

        elif self.tool == "eraser":
            item       =       self.canvas.create_line(self.last_x,
self.last_y, event.x, event.y,
```

```
                                            fill="white",
width=self.brush_size,
                                            capstyle=tk.ROUND,
smooth=True)
            self.undo_stack.append(item)
            self.last_x, self.last_y = event.x, event.y

        elif self.tool in ["line", "rect", "oval"]:
            if self.temp_shape:
                self.canvas.delete(self.temp_shape)
            if self.tool == "line":
                self.temp_shape                                    =
self.canvas.create_line(self.last_x, self.last_y, event.x, event.y,

fill=self.brush_color, width=self.brush_size)
            elif self.tool == "rect":
                self.temp_shape                                    =
self.canvas.create_rectangle(self.last_x,   self.last_y,   event.x,
event.y,

outline=self.brush_color, width=self.brush_size)
            elif self.tool == "oval":
                self.temp_shape                                    =
self.canvas.create_oval(self.last_x, self.last_y, event.x, event.y,

outline=self.brush_color, width=self.brush_size)

    def reset(self, event):
        if   self.tool   in   ["line",   "rect",   "oval"]   and
self.temp_shape:
            coords = self.canvas.coords(self.temp_shape)
            self.canvas.delete(self.temp_shape)
            if time.time() - self.start_time > 1:
                item    =    self.canvas.create_rectangle(*coords,
fill=self.brush_color)
            else:
                if self.tool == "line":
                    item    =    self.canvas.create_line(*coords,
fill=self.brush_color, width=self.brush_size)
                elif self.tool == "rect":
```

```
                    item = self.canvas.create_rectangle(*coords,
outline=self.brush_color, width=self.brush_size)
                elif self.tool == "oval":
                    item = self.canvas.create_oval(*coords,
outline=self.brush_color, width=self.brush_size)
            self.undo_stack.append(item)
            self.temp_shape = None
        elif self.tool == "text":
            text = simpledialog.askstring("Input Text", "Masukkan
teks:")
            if text:
                item = self.canvas.create_text(event.x, event.y,
text=text, fill=self.brush_color, font=("Arial", 14), anchor=tk.NW)
                self.undo_stack.append(item)

        self.last_x, self.last_y = None, None

    def save_canvas(self):
        self.root.update()
        x = self.canvas.winfo_rootx()
        y = self.canvas.winfo_rooty()
        x1 = x + self.canvas.winfo_width()
        y1 = y + self.canvas.winfo_height()
        filepath                                                =
filedialog.asksaveasfilename(defaultextension=".png",
filetypes=[("PNG files", "*.png")])
        if filepath:
            ImageGrab.grab().crop((x, y, x1, y1)).save(filepath)
            print(f"Saved to {filepath}")

if __name__ == "__main__":
    root = tk.Tk()
    app = DrawingApp(root)
    root.mainloop()
```

Tabel 1. *Source Code*

4.2 Pembahasan

Aplikasi *Advanced Drawing App* yang telah berhasil dikembangkan memiliki beberapa aspek teknis dan fungsional yang menarik untuk dibahas, terutama dari sisi implementasi fitur, interaksi pengguna, serta efisiensi struktur program. Berikut ini pembahasan berdasarkan fitur-fitur utama:

4.2.1 Antarmuka Pengguna yang Intuitif

Antarmuka aplikasi didesain secara sederhana namun fungsional. Seluruh alat gambar ditampilkan dalam bentuk ikon pada toolbar bagian atas. Pemilihan ikon dibanding teks dinilai memudahkan pengguna dalam mengenali fungsi tanpa perlu membaca secara detail. Slider pengatur ketebalan juga ditempatkan di toolbar untuk akses cepat. Letak toolbar yang terpusat di bagian atas memungkinkan pengguna berfokus pada kanvas sebagai area utama interaksi.

4.2.2 Respons Interaksi Mouse dan Event-Driven Programming

Aplikasi ini menerapkan model pemrograman berbasis event (event-driven), di mana setiap aksi pengguna—seperti klik, drag, dan release—ditangani oleh event handler khusus. Fungsi start_draw(), draw(), dan reset() saling berkaitan untuk membentuk satu rangkaian aksi menggambar. Hal ini memberikan pengalaman interaktif yang responsif saat menggambar bebas maupun saat membentuk objek geometris.

4.2.3 Fleksibilitas Tool Gambar

Berbagai alat menggambar berhasil diimplementasikan, antara lain:

- Pencil Tool: Menghasilkan garis halus sesuai pergerakan mouse.
- Crayon Tool: Menggunakan efek stipple dan random offset untuk meniru tekstur krayon.
- Eraser Tool: Menggambar warna putih untuk menyamarkan gambar sebelumnya.
- Line, Rectangle, Oval: Dibuat dengan metode klik dan drag. Gambar sementara akan ditampilkan selama drag, kemudian diganti dengan gambar permanen saat mouse dilepas.
- Text Tool: Menggunakan simpledialog untuk menerima input teks dari pengguna dan meletakkannya langsung di canvas.

Fitur-fitur ini mencerminkan kemampuan manipulasi objek grafis secara langsung oleh pengguna melalui antarmuka visual yang mudah diakses.

4.2.4 Undo dan Redo Berbasis Stack

Fitur undo dan redo bekerja menggunakan dua stack: undo_stack dan redo_stack. Setiap objek yang digambar di canvas memiliki ID yang disimpan dalam stack. Fungsi undo akan menghapus objek terakhir dan memindahkannya ke redo_stack, sedangkan redo mengembalikan objek tersebut ke canvas. Mekanisme ini cukup efektif untuk pengelolaan aksi pengguna, meskipun masih bersifat linear dan belum mendukung grouping objek.

4.2.5 Penyimpanan Gambar sebagai PNG

Penyimpanan gambar dilakukan menggunakan pustaka Pillow melalui fungsi ImageGrab.grab(). Proses ini mengambil snapshot area kanvas berdasarkan koordinat posisi widget di layar. Gambar hasil tangkapan kemudian dipotong sesuai ukuran kanvas dan disimpan dalam format PNG.

Contoh penggunaan:

```
ImageGrab.grab().crop((x, y, x1, y1)).save(filepath)
```

Fitur ini memastikan pengguna dapat menyimpan hasil karya dengan kualitas gambar tinggi dan kompatibel dengan berbagai aplikasi lain.

4.2.6 Dokumentasi Hasil Gambar

Pada Gambar 1 ditunjukkan hasil implementasi berbagai fitur:

- Teks “Hai, Windy” ditulis menggunakan text tool.
- Goresan pensil dan garis bawah menggunakan pencil tool.
- Persegi berwarna merah menggunakan shape rectangle.
- Teks dengan efek krayon berwarna merah.
- Bentuk oval menggunakan shape oval dengan warna dan ketebalan yang telah diatur sebelumnya.

Gambar tersebut mencerminkan keberhasilan integrasi seluruh fitur dalam satu sesi penggunaan aplikasi.

4.2.7 Evaluasi Kelebihan dan Kekurangan

Kelebihan:

- Antarmuka sederhana dan mudah digunakan.
- Fitur lengkap untuk aplikasi edukatif menggambar digital.
- Kode modular dan mudah dikembangkan lebih lanjut.

Kekurangan:

- Belum mendukung layer management.
- Undo/redo belum menyertakan pengelompokan tindakan.
- Tidak terdapat fitur zoom atau grid untuk membantu ketelitian menggambar.

BAB V
PENUTUP

5.1 Kesimpulan

Berdasarkan hasil perancangan, implementasi, dan pengujian terhadap aplikasi *Advanced Drawing App*, dapat disimpulkan bahwa:

1. Aplikasi berhasil dibangun menggunakan bahasa Python dengan pustaka Tkinter dan Pillow sebagai dasar pembentukan antarmuka grafis serta penyimpanan gambar.
2. Fitur-fitur utama seperti pensil, crayon, penghapus, bentuk geometris (garis, persegi, dan oval), input teks, pengaturan warna, slider ketebalan kuas, undo/redo, clear canvas, dan penyimpanan gambar dalam format PNG telah berfungsi dengan baik dan sesuai tujuan awal perancangan.
3. Antarmuka pengguna (UI) dirancang secara sederhana, intuitif, dan dapat digunakan tanpa pelatihan khusus, sehingga cocok digunakan oleh pemula maupun pelajar yang sedang belajar tentang grafika komputer.
4. Teknik manajemen objek berbasis *stack* pada fitur undo/redo serta penggunaan event-handler mouse memberikan interaktivitas yang baik terhadap pengguna dan mempermudah pengelolaan proses menggambar.
5. Aplikasi ini telah memenuhi fungsi sebagai media edukatif untuk memahami konsep dasar grafika komputer, manipulasi objek visual, serta interaksi GUI dalam konteks pemrograman Python.

5.2 Saran

Agar aplikasi *Advanced Drawing App* dapat dikembangkan lebih lanjut dan memiliki nilai tambah yang lebih tinggi, berikut beberapa saran pengembangan:

1. **Penambahan fitur layer** agar pengguna dapat mengatur elemen gambar secara terpisah dan mengelola urutan objek dengan lebih fleksibel.
2. **Dukungan zoom dan grid**, untuk memudahkan pengguna dalam menggambar secara detail dan presisi tinggi.
3. **Mekanisme undo/redo yang lebih kompleks**, seperti penyimpanan snapshot atau riwayat aksi berbasis waktu, sehingga dapat mengelola penghapusan grup objek sekaligus.

4. **Penyimpanan proyek** dalam format khusus (misal JSON atau SVG) agar gambar dapat disimpan dan dilanjutkan kembali di lain waktu.
5. **Responsivitas terhadap berbagai ukuran layar**, serta kemampuan *resize* canvas secara dinamis agar aplikasi dapat dijalankan pada berbagai resolusi perangkat.

DAFTAR PUSTAKA

Python Software Foundation. (2023). *Python 3.11.3 documentation*. https://docs.python.org/3/

Lundh, F. (2005). *An introduction to Tkinter*. Pythonware. http://effbot.org/tkinterbook/

Pillow Developers. (2023). *Pillow (PIL Fork) documentation*. https://pillow.readthedocs.io/

Grayson, J. E. (2000). *Python and Tkinter programming*. Manning Publications.

Zelle, J. M. (2004). *Python programming: An introduction to computer science* (2nd ed.). Franklin, Beedle & Associates.

Norton, P. (2020). *Programming fundamentals using Python*. Wiley.

Guttag, J. V. (2016). *Introduction to computation and programming using Python: With application to understanding data* (2nd ed.). MIT Press.

LAMPIRAN

Link Github : https://github.com/windyclaun/Aplikasi-Paint-Sederhana.git