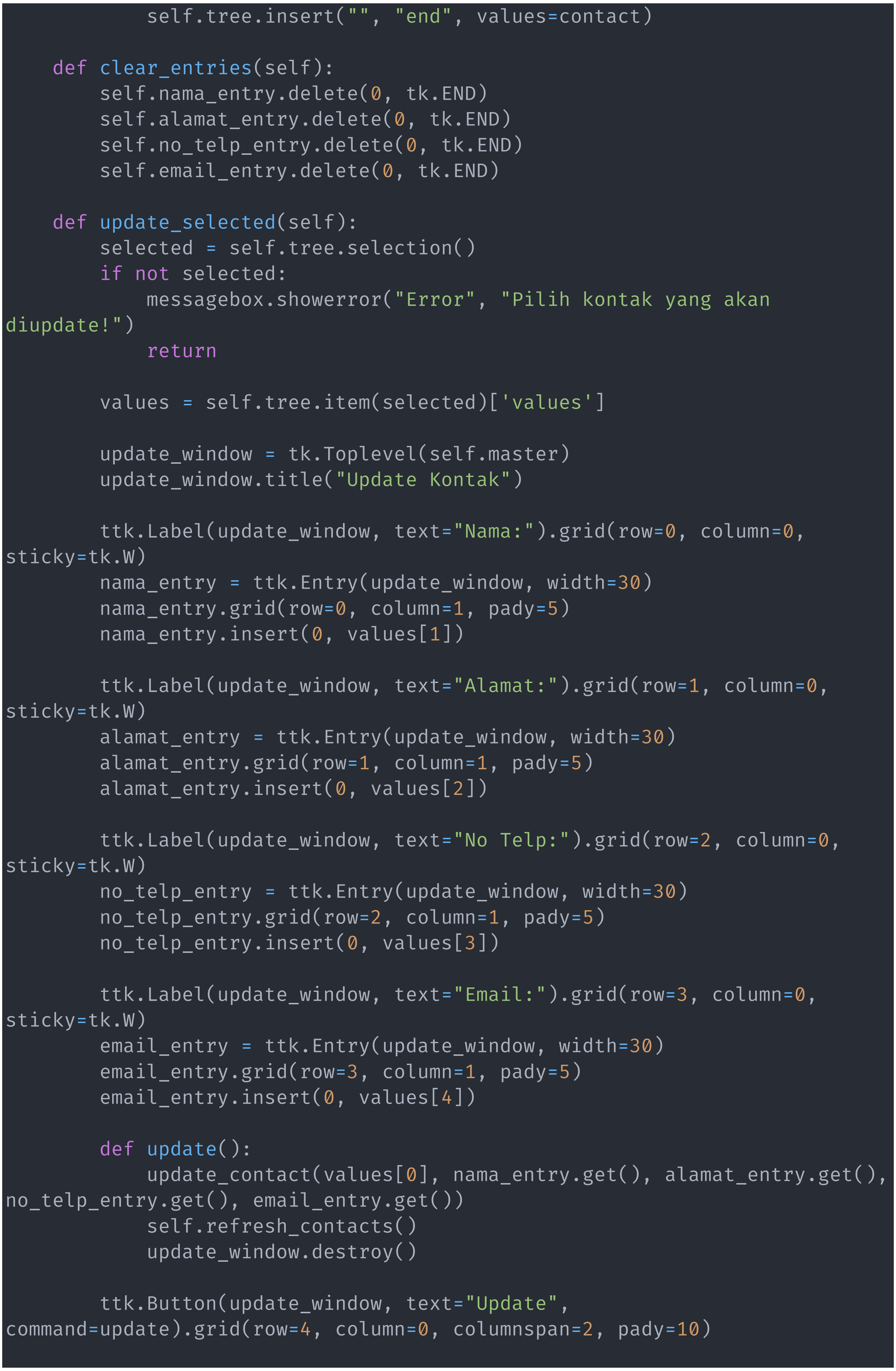
1. Prerequisite: Membuat Database dan Tabel

```sql
CREATE DATABASE address_book;
USE address_book;

CREATE TABLE contacts (
    id INT AUTO_INCREMENT PRIMARY KEY,
    nama VARCHAR(100),
    alamat TEXT,
    no_telp VARCHAR(20),
    email VARCHAR(100)
);
```

2. Implementasi CRUD dengan Python dan MySQL Connector

```python
import mysql.connector
import tkinter as tk
from tkinter import ttk, messagebox

# Koneksi ke database
def create_connection():
    return mysql.connector.connect(
        host="localhost",
        user="your_username",
        password="your_password",
        database="address_book"
    )

# CRUD #1 - Create
def insert_contact(nama, alamat, no_telp, email):
    conn = create_connection()
    cursor = conn.cursor()
    query = "INSERT INTO contacts (nama, alamat, no_telp, email) VALUES (%s, %s, %s, %s)"
    values = (nama, alamat, no_telp, email)
    cursor.execute(query, values)
    conn.commit()
    cursor.close()
    conn.close()

# CRUD #2 - Retrieve
def get_contacts():
    conn = create_connection()
    cursor = conn.cursor()
    query = "SELECT * FROM contacts"
    cursor.execute(query)
    contacts = cursor.fetchall()
    cursor.close()
    conn.close()
    return contacts

# CRUD #3 - Update
```

```
def update_contact(id, nama, alamat, no_telp, email):
    conn = create_connection()
    cursor = conn.cursor()
    query = "UPDATE contacts SET nama=%s, alamat=%s, no_telp=%s, email=%s
WHERE id=%s"
    values = (nama, alamat, no_telp, email, id)
    cursor.execute(query, values)
    conn.commit()
    cursor.close()
    conn.close()

# CRUD #4 - Delete
def delete_contact(id):
    conn = create_connection()
    cursor = conn.cursor()
    query = "DELETE FROM contacts WHERE id=%s"
    values = (id,)
    cursor.execute(query, values)
    conn.commit()
    cursor.close()
    conn.close()

# GUI dengan Tkinter
class AddressBookApp:
    def __init__(self, master):
        self.master = master
        self.master.title("Buku Alamat")
        self.master.geometry("600x400")

        self.create_widgets()
        self.refresh_contacts()

    def create_widgets(self):
        # Frame untuk input
        input_frame = ttk.Frame(self.master, padding="10")
        input_frame.grid(row=0, column=0, sticky=(tk.W, tk.E, tk.N, tk.S))

        ttk.Label(input_frame, text="Nama:").grid(row=0, column=0,
sticky=tk.W)
        self.nama_entry = ttk.Entry(input_frame, width=30)
        self.nama_entry.grid(row=0, column=1, pady=5)

        ttk.Label(input_frame, text="Alamat:").grid(row=1, column=0,
sticky=tk.W)
        self.alamat_entry = ttk.Entry(input_frame, width=30)
        self.alamat_entry.grid(row=1, column=1, pady=5)

        ttk.Label(input_frame, text="No Telp:").grid(row=2, column=0,
sticky=tk.W)
        self.no_telp_entry = ttk.Entry(input_frame, width=30)
        self.no_telp_entry.grid(row=2, column=1, pady=5)
        ttk.Label(input_frame, text="Email:").grid(row=3, column=0,
sticky=tk.W)
        self.email_entry = ttk.Entry(input_frame, width=30)
        self.email_entry.grid(row=3, column=1, pady=5)

        ttk.Button(input_frame, text="Tambah Kontak",
command=self.add_contact).grid(row=4, column=0, columnspan=2, pady=10)

        # Treeview untuk menampilkan kontak
        self.tree = ttk.Treeview(self.master, columns=("ID", "Nama",
"Alamat", "No Telp", "Email"), show="headings")
        self.tree.grid(row=1, column=0, sticky=(tk.W, tk.E, tk.N, tk.S))

        self.tree.heading("ID", text="ID")
        self.tree.heading("Nama", text="Nama")
        self.tree.heading("Alamat", text="Alamat")
        self.tree.heading("No Telp", text="No Telp")
        self.tree.heading("Email", text="Email")

        # Scrollbar
        scrollbar = ttk.Scrollbar(self.master, orient=tk.VERTICAL,
command=self.tree.yview)
        scrollbar.grid(row=1, column=1, sticky=(tk.N, tk.S))
        self.tree.configure(yscrollcommand=scrollbar.set)

        # Tombol Update dan Delete
        button_frame = ttk.Frame(self.master, padding="10")
        button_frame.grid(row=2, column=0, sticky=(tk.W, tk.E))

        ttk.Button(button_frame, text="Update",
command=self.update_selected).grid(row=0, column=0, padx=5)
        ttk.Button(button_frame, text="Delete",
command=self.delete_selected).grid(row=0, column=1, padx=5)

    def add_contact(self):
        nama = self.nama_entry.get()
        alamat = self.alamat_entry.get()
        no_telp = self.no_telp_entry.get()
        email = self.email_entry.get()

        if nama and alamat and no_telp and email:
            insert_contact(nama, alamat, no_telp, email)
            self.refresh_contacts()
            self.clear_entries()
        else:
            messagebox.showerror("Error", "Semua field harus diisi!")

    def refresh_contacts(self):
        for i in self.tree.get_children():
            self.tree.delete(i)

        contacts = get_contacts()
        for contact in contacts:
```



```
            self.tree.insert("", "end", values=contact)

    def clear_entries(self):
        self.nama_entry.delete(0, tk.END)
        self.alamat_entry.delete(0, tk.END)
        self.no_telp_entry.delete(0, tk.END)
        self.email_entry.delete(0, tk.END)

    def update_selected(self):
        selected = self.tree.selection()
        if not selected:
            messagebox.showerror("Error", "Pilih kontak yang akan
diupdate!")
            return

        values = self.tree.item(selected)['values']

        update_window = tk.Toplevel(self.master)
        update_window.title("Update Kontak")

        ttk.Label(update_window, text="Nama:").grid(row=0, column=0,
sticky=tk.W)
        nama_entry = ttk.Entry(update_window, width=30)
        nama_entry.grid(row=0, column=1, pady=5)
        nama_entry.insert(0, values[1])

        ttk.Label(update_window, text="Alamat:").grid(row=1, column=0,
sticky=tk.W)
        alamat_entry = ttk.Entry(update_window, width=30)
        alamat_entry.grid(row=1, column=1, pady=5)
        alamat_entry.insert(0, values[2])

        ttk.Label(update_window, text="No Telp:").grid(row=2, column=0,
sticky=tk.W)
        no_telp_entry = ttk.Entry(update_window, width=30)
        no_telp_entry.grid(row=2, column=1, pady=5)
        no_telp_entry.insert(0, values[3])

        ttk.Label(update_window, text="Email:").grid(row=3, column=0,
sticky=tk.W)
        email_entry = ttk.Entry(update_window, width=30)
        email_entry.grid(row=3, column=1, pady=5)
        email_entry.insert(0, values[4])

        def update():
            update_contact(values[0], nama_entry.get(), alamat_entry.get(),
no_telp_entry.get(), email_entry.get())
            self.refresh_contacts()
            update_window.destroy()

        ttk.Button(update_window, text="Update",
command=update).grid(row=4, column=0, columnspan=2, pady=10)
```

```
    def delete_selected(self):
        selected = self.tree.selection()
        if not selected:
            messagebox.showerror("Error", "Pilih kontak yang akan
dihapus!")
            return

        if messagebox.askyesno("Konfirmasi", "Anda yakin ingin menghapus
kontak ini?"):
            values = self.tree.item(selected)['values']
            delete_contact(values[0])
            self.refresh_contacts()

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