**LAPORAN APLIKASI PEMESANAN TIKET BIOSKOP**

**PEMROGRAMAN BERORIENTASI OBJEK**

**KELAS VIII**

T. A. Semester Ganjil 2024/2025

****

**5230411283 Firman Hasibuan**

**5230411308 Muhammad Aulia Rahman**

**5230411294 Fradeka Nur Choerun Kristiyanto**

**5230411329 Mochammad Adi Prawira**

**PROGRAM STUDI INFORMATIKA**

**FAKULTAS SAINS & TEKNOLOGI**

**UNIVERSITAS TEKNOLOGI YOGYAKARTA**

**YOGYAKARTA**

**BAB I**

**PENDAHULUAN**

1. **Latar Belakang**

Perkembangan teknologi informasi dan komunikasi telah mencapai perubahan yang signifikan dalam berbagai aspek kehidupan, terutama dalam industri hiburan. Salah satu bentuk hiburan yang paling populer adalah menonton film di bioskop. Dengan meningkatnya minat masyarakat untuk menonton film, kebutuhan akan sistem pemesanan tiket bioskop yang efisien menjadi semakin penting. Banyak pelanggan seringkali harus mengantri lama untuk mendapatkan tiket, terutama saat film-film baru yang sangat dinanti ditayangkan.

Untuk mengatasi masalah ini, kami mengembangkan sebuah aplikasi pemesanan tiket bioskop berbasis GUI (Graphical User Interface) yang menerapkan prinsip CRUD (Create, Read, Update, Delete) menjadi sangat relevan. Aplikasi ini dirancang untuk mempermudah pengguna dalam melakukan pemesanan tiket secara online dengan antarmuka yang menarik. Dengan sistem ini, pengguna dapat:

* **Memilih film dan waktu tayang**: Pengguna dapat melihat daftar film yang tayang beserta jadwalnya.
* **Memilih tempat duduk**: Aplikasi memungkinkan pengguna untuk memilih posisi tempat duduk yang diinginkan dari denah yang tersedia.

**BAB II**

**PEMBAHASAN**

1. **Diagram**

**A diagram of a flowchart

Description automatically generated**

Berikut merupakan rancangan diagram dari program yang dibuat. Menu utama dari aplikasi yang akan dibuat adalah menu pemesanan tiket. Dan pada diagram ini menggunakan Mulai, Input, Data, Decision, dan Process. Dan terdapat juga class untuk menginput dimana inputan dimasukkan secara manual pada program yang dijalankan.

1. **implementasi running program dan penjelasan fitur**

A screenshot of a computer screen

Description automatically generated

program ini dibuat dengan menggunakan code Bahasa python dengan tujuan untuk dapat tersambung dengan database mySQL. Selain terdapat code program untuk menyambungkan database terdapat code program untuk memasukan data dari terminal langsung ke database yang telah dibuat. Fungsi utamanya adalah untuk mengelola data bioskop yang terdiri dari tabel seperti tambah film, pesan tiket, dan struk.

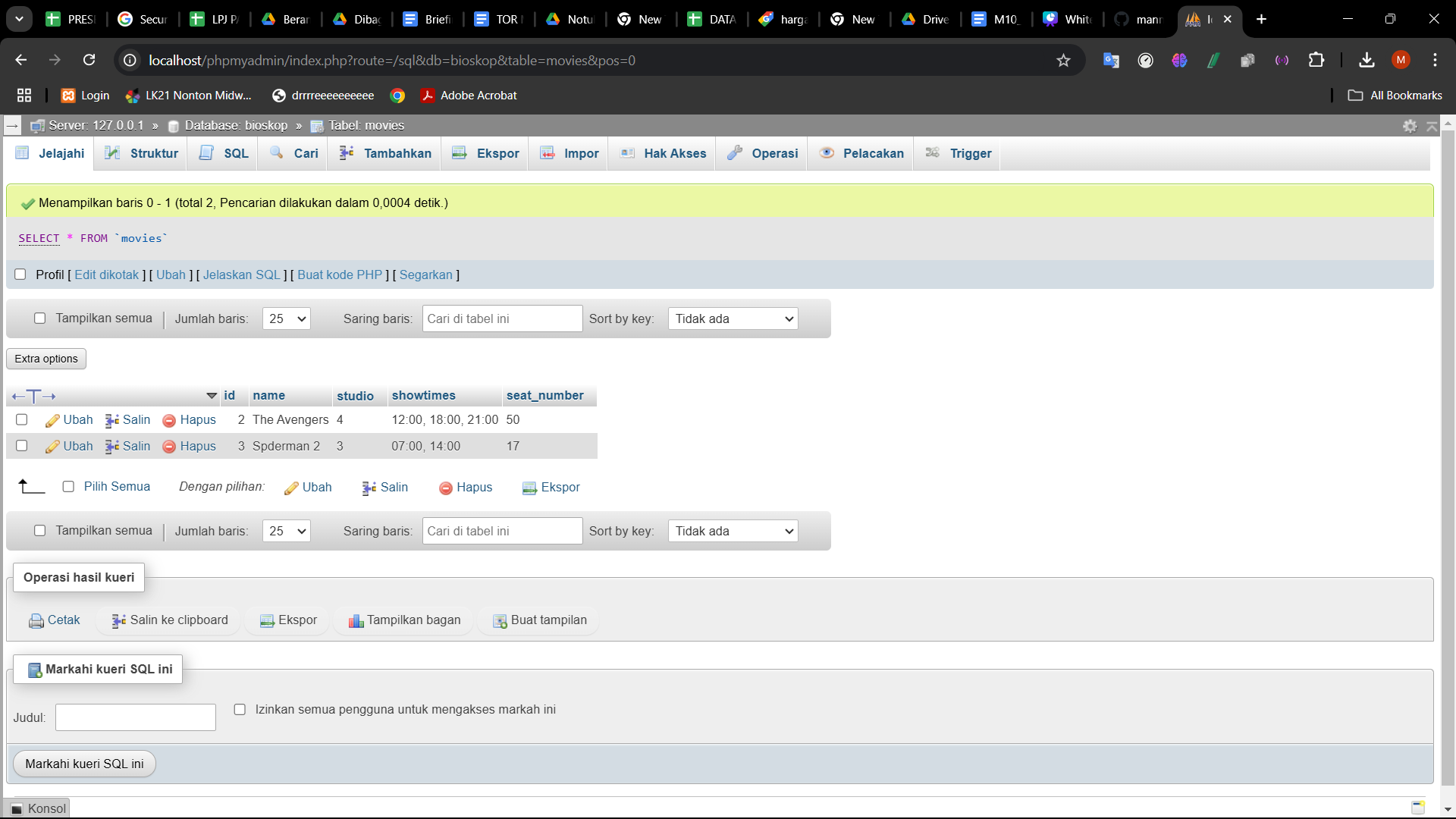
1. **Fitur Utama**

* Membuat database bioskop dan tabel film
* Menampilkan semua data film dalam format tabel.
* Menambah, mengedit, dan menghapus data film melalui jendela GUI tambahan.
* Memilih film, jadwal tayang, dan kursi untuk mencetak tiket bioskop
* Menampilkan tiket yang dicetak dalam format khusus.

1. **Penjelasan Komponen GUI**

* Menu Bar: Memberikan opsi untuk melihat semua data film, keluar dari aplikasi, atau mengelola data film.
* Pilihan Film: Pengguna dapat memilih film dari daftar yang tersedia. Daftar ini diambil langsung dari database
* Pemilihan Jadwal dan Kursi: Berdasarkan film yang dipilih, pengguna bisa memilih jadwal tayang dan kursi.
* Cetak Tiket: Setelah semua pilihan diisi, aplikasi akan mencetak tiket dalam format teks yang ditampilkan di layar.
* Menyediakan menu berbasis Python tkinter GUI

1. **Penjelasan Struktur Tabel**



Tabel yang digunakan pada aplikasi ini ada satu. Yaitu tabel movies untuk menambah film, jadwal film, studio film, dan jumlah kursi dengan primary key id

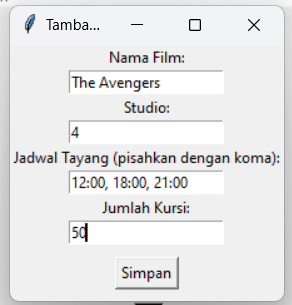
1. **Penjelasan Alur Aplikasi**
2. **Inisialisasi Database**

Pada saat aplikasi dijalankan, tabel-tabel yang dibutuhkan dibuat secara otomatis (jika belum ada).

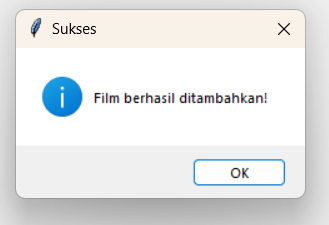
1. **Pengelolaan Data Film (CRUD)**

Pengguna dapat menambah, memperbarui, atau menghapus data film melalui antarmuka GUI yang intuitif.

1. **Tambah Film**



Disini adalah fitur menambakan film yang akan tayang, Studio, Jadwal tayang, dan jawal kursi yang tersedia pada film di studio tersebut.



1. **Tampil Data Film**

A screenshot of a computer screen

Description automatically generated

A screenshot of a movie

Description automatically generated

Terdapat fitur tampil data film yang akan tayang di bioskop

1. **Edit data Film**

A screenshot of a computer screen

Description automatically generated

Terdapat fitur untuk mengedit data film ketika film, studio, jadwal tayang, dan jumlah kursi salah menginput.

A screenshot of a computer

Description automatically generated

1. **Hapus Film**

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

Terdapat fitur untuk menghapus film dari data yang tersedia.

1. **Pemesanan Tiket**

Pengguna dapat memilih film, jadwal tayang, dan kursi. Sistem memverifikasi input pengguna dan menghasilkan struk tiket bioskop.

A screenshot of a computer

Description automatically generated

1. **kode program**

import mysql.connector

conn = mysql.connector.connect(

    user = "root",

    host = "localhost",

    password = "",

    database = "penjualan"

)

cur = conn.cursor()

#membuat database

# cur.execute("CREATE DATABASE pejualan")

cur.execute('DROP TABLE IF EXISTS Struk')

cur.execute('DROP TABLE IF EXISTS Transaksi')

cur.execute('DROP TABLE IF EXISTS Produk')

cur.execute('DROP TABLE IF EXISTS Pegawai')

# # Membuat tabel Pegawai

cur.execute("""

CREATE TABLE IF NOT EXISTS Pegawai (

    NIK VARCHAR(7) NOT NULL PRIMARY KEY,

    Nama VARCHAR(25),

    Alamat VARCHAR(225)

)

""")

# # Membuat tabel Produk

cur.execute("""

CREATE TABLE IF NOT EXISTS Produk (

    Kode\_Produk CHAR(10) NOT NULL PRIMARY KEY,

    Nama\_Produk VARCHAR(10),

    Jenis\_produk VARCHAR(225),

    Harga DECIMAL(10, 2)

)

""")

# # Membuat tabel Transaksi

cur.execute("""

CREATE TABLE IF NOT EXISTS Transaksi (

    No\_Transaksi INT NOT NULL PRIMARY KEY,

    Detail\_Transaksi CHAR(10) NOT NULL,

    NIK VARCHAR(7),

    FOREIGN KEY (NIK) REFERENCES Pegawai(NIK)

)

""")

# # Membuat tabel Struk

cur.execute("""

CREATE TABLE IF NOT EXISTS Struk (

    No\_Transaksi INT NOT NULL PRIMARY KEY,

    Nama\_pegawai VARCHAR(25),

    Nama\_Produk VARCHAR(25),

    Kode\_Produk CHAR(10),

    Jumlah\_Produk INT(15),

    Total\_Harga DECIMAL(10, 2),

    FOREIGN KEY (Kode\_Produk) REFERENCES Produk(Kode\_Produk),

    FOREIGN KEY (No\_Transaksi) REFERENCES Transaksi(No\_Transaksi)

)

""")

# Membuat menu utama

def tampil\_data():

    tabel = input("Masukkan nama tabel (Pegawai/Produk/Transaksi/Struk): ")

    cur.execute(f"SELECT \* FROM {tabel}")

    rows = cur.fetchall()

    for row in rows:

        print(row)

def input\_semua\_data():

    # Input data untuk semua tabel

    tambah\_pegawai()

    tambah\_produk()

    tambah\_transaksi()

    tambah\_struk()

def input\_data\_tertentu():

    tabel = input("Masukkan nama tabel untuk input data (Pegawai/Produk/Transaksi/Struk): ")

    if tabel.lower() == "pegawai":

        tambah\_pegawai()

    elif tabel.lower() == "produk":

        tambah\_produk()

    elif tabel.lower() == "transaksi":

        tambah\_transaksi()

    elif tabel.lower() == "struk":

        tambah\_struk()

    else:

        print("Tabel tidak valid.")

def ubah\_data():

    tabel = input("Masukkan nama tabel untuk ubah data (Pegawai/Produk/Transaksi/Struk): ")

    kolom = input("Masukkan kolom yang ingin diubah: ")

    nilai\_baru = input("Masukkan nilai baru: ")

    kondisi = input("Masukkan kondisi (contoh: NIK='123'): ")

    cur.execute(f"UPDATE {tabel} SET {kolom} = '{nilai\_baru}' WHERE {kondisi}")

    conn.commit()

    print("Data berhasil diubah.")

def hapus\_data():

    tabel = input("Masukkan nama tabel untuk hapus data (Pegawai/Produk/Transaksi/Struk): ")

    kondisi = input("Masukkan kondisi (contoh: NIK='123'): ")

    cur.execute(f"DELETE FROM {tabel} WHERE {kondisi}")

    conn.commit()

    print("Data berhasil dihapus.")

def tambah\_pegawai():

    nik = input("Masukkan NIK: ")

    cur.execute("SELECT \* FROM Pegawai WHERE NIK = %s", (nik,))

    if cur.fetchone():

        print("NIK sudah ada dalam database. Silakan gunakan NIK lain atau perbarui data yang sudah ada.")

    else:

        nama = input("Masukkan Nama: ")

        alamat = input("Masukkan Alamat: ")

        cur.execute("INSERT INTO Pegawai (NIK, Nama, Alamat) VALUES (%s, %s, %s)", (nik, nama, alamat))

        conn.commit()

        print("Data Pegawai berhasil ditambahkan.")

def tambah\_produk():

    kode\_produk = input("Masukkan Kode Produk: ")

    nama\_produk = input("Masukkan Nama Produk: ")

    jenis\_produk = input("Masukkan Jenis Produk: ")

    harga = float(input("Masukkan Harga: "))

    cur.execute("INSERT INTO Produk (Kode\_Produk, Nama\_Produk, Jenis\_produk, Harga) VALUES (%s, %s, %s, %s)", (kode\_produk, nama\_produk, jenis\_produk, harga))

    conn.commit()

    print("Data Produk berhasil ditambahkan.")

def tambah\_transaksi():

    no\_transaksi = int(input("Masukkan Nomor Transaksi: "))

    detail\_transaksi = input("Masukkan Detail Transaksi (Kode Produk): ")

    nik = input("Masukkan NIK Pegawai: ")

    cur.execute("INSERT INTO Transaksi (No\_Transaksi, Detail\_Transaksi, NIK) VALUES (%s, %s, %s)", (no\_transaksi, detail\_transaksi, nik))

    conn.commit()

    print("Data Transaksi berhasil ditambahkan.")

def tambah\_struk():

    no\_transaksi = int(input("Masukkan Nomor Transaksi: "))

    nama\_pegawai = input("Masukkan Nama Pegawai: ")

    nama\_produk = input("Masukkan Nama Produk: ")

    jumlah\_produk = int(input("Masukkan Jumlah Produk: "))

    total\_harga = float(input("Masukkan Total Harga: "))

    cur.execute("INSERT INTO Struk (NoTransaksi, Nama\_pegawai, Nama\_Produk, Jumlah\_Produk, Total\_Harga) VALUES (%s, %s, %s, %s, %s)", (no\_transaksi, nama\_pegawai, nama\_produk, jumlah\_produk, total\_harga))

    conn.commit()

    print("Data Struk berhasil ditambahkan.")

# Menu utama

while True:

    print("\nMenu:")

    print("1. Tampil Data")

    print("2. Input Semua Data")

    print("3. Input Data Tertentu")

    print("4. Ubah Data")

    print("5. Hapus Data")

    print("6. Keluar")

    menu = input("Pilihan Menu: ")

    if menu == "1":

        tampil\_data()

    elif menu == "2":

        input\_semua\_data()

    elif menu == "3":

        input\_data\_tertentu()

    elif menu == "4":

        ubah\_data()

    elif menu == "5":

        hapus\_data()

    elif menu == "6":

        print("Keluar dari program.")

        break

    else:

        print("Pilihan tidak valid.")

import tkinter as tk

from tkinter import ttk, messagebox

from datetime import datetime

import mysql.connector

# Inisialisasi database

def init\_db():

conn = mysql.connector.connect(

host="localhost",

user="root",

password="",

database="bioskop"

)

cur = conn.cursor()

# Drop table if exists and create table

cur.execute("DROP TABLE IF EXISTS movies")

cur.execute("""

CREATE TABLE movies (

id INT AUTO\_INCREMENT PRIMARY KEY,

name VARCHAR(255) NOT NULL,

studio VARCHAR(255),

showtimes VARCHAR(255),

seat\_number VARCHAR(50)

)

""")

conn.commit()

cur.close()

conn.close()

# Fungsi untuk menampilkan seluruh data film

def show\_all\_movies():

conn = mysql.connector.connect(

host="localhost",

user="root",

password="",

database="bioskop"

)

cursor = conn.cursor()

cursor.execute('SELECT \* FROM movies')

movies = cursor.fetchall()

conn.close()

# Menampilkan data dalam messagebox

movies\_text = "\n".join([f"id: {movie[0]},Name: {movie[1]}, Studio: {movie[2]}, Showtimes: {movie[3]}, Seats: {movie[4]}" for movie in movies])

messagebox.showinfo("Daftar Film", movies\_text if movies else "Tidak ada data film.")

# Fungsi untuk menambah data film

def add\_movie(name, studio, showtimes, seats):

conn = mysql.connector.connect(

host="localhost",

user="root",

password="",

database="bioskop"

)

cursor = conn.cursor()

cursor.execute('''

INSERT INTO movies (name, studio, showtimes, seat\_number)

VALUES (%s, %s, %s, %s)

''', (name, studio, showtimes, seats))

conn.commit()

conn.close()

# Fungsi untuk menghapus data film berdasarkan ID

def delete\_movie(movie\_id):

conn = mysql.connector.connect(

host="localhost",

user="root",

password="",

database="bioskop"

)

cursor = conn.cursor()

cursor.execute('DELETE FROM movies WHERE id = %s', (movie\_id,))

conn.commit()

cursor.close()

conn.close()

# Fungsi untuk mengupdate data film

def update\_movie(id, name, studio, showtimes, seats):

conn = mysql.connector.connect(

host="localhost",

user="root",

password="",

database="bioskop"

)

cursor = conn.cursor()

cursor.execute('''

UPDATE movies

SET name = %s, studio = %s, showtimes = %s, seat\_number = %s

WHERE id = %s

''', (name, studio, showtimes, seats, id,))

conn.commit()

conn.close()

class TicketApp:

def \_\_init\_\_(self, root):

self.root = root

self.root.title("Aplikasi Tiket Bioskop")

self.root.geometry("500x500")

# Inisialisasi database

init\_db()

self.widget\_create()

def widget\_create(self):

# Menu bar

menubar = tk.Menu(self.root)

self.root.config(menu=menubar)

# Menambahkan menu

file\_menu = tk.Menu(menubar, tearoff=0)

menubar.add\_cascade(label="Menu", menu=file\_menu)

file\_menu.add\_command(label="Tampilkan Semua Data", command=show\_all\_movies)

file\_menu.add\_separator()

file\_menu.add\_command(label="Keluar", command=self.root.quit)

manage\_menu = tk.Menu(menubar, tearoff=0)

menubar.add\_cascade(label="Kelola Data Film", menu=manage\_menu)

manage\_menu.add\_command(label="Tambah Data Film", command=self.add\_movie\_window)

manage\_menu.add\_command(label="Edit Data Film", command=self.edit\_movie\_window)

manage\_menu.add\_command(label="Hapus Data Film", command=self.delete\_movie\_window)

# Label judul

title\_label = tk.Label(self.root, text="Aplikasi Tiket Bioskop", font=("Arial", 16))

title\_label.pack(pady=10)

# Pilihan film

self.movie\_label = tk.Label(self.root, text="Pilih Film:")

self.movie\_label.pack()

self.movie\_combobox = ttk.Combobox(self.root)

self.movie\_combobox.pack()

self.movie\_combobox.bind("<<ComboboxSelected>>", self.update\_showtimes)

# Pemilihan jadwal film

self.showtime\_label = tk.Label(self.root, text="Pilih Jadwal Tayang:")

self.showtime\_label.pack()

self.showtime\_combobox = ttk.Combobox(self.root)

self.showtime\_combobox.pack()

self.showtime\_combobox.bind("<<ComboboxSelected>>", self.update\_showtimes)

# Pemilihan kursi bioskop

self.seat\_label = tk.Label(self.root, text="Pilih Kursi:")

self.seat\_label.pack()

self.seat\_spinbox = tk.Spinbox(self.root, from\_=1, to=50)

self.seat\_spinbox.pack()

# Tombol untuk menghasilkan tiket

self.generate\_button = tk.Button(self.root, text="Cetak Tiket", command=self.generate\_ticket)

self.generate\_button.pack(pady=10)

# Menampilkan tiket

self.ticket\_display = tk.Label(self.root, text="", justify="left", font=("Arial", 12), bg="lightgray", relief="sunken", padx=10, pady=10)

self.ticket\_display.pack(pady=15)

self.update\_movie\_combobox() # Memperbarui daftar film

def update\_showtimes(self, event):

selected\_movie = self.movie\_combobox.get()

conn = mysql.connector.connect(

host="localhost",

user="root",

password="",

database="bioskop"

)

cursor = conn.cursor()

cursor.execute('SELECT showtimes FROM movies WHERE name = %s', (selected\_movie,))

showtimes = cursor.fetchone()[0].split(",")

conn.close()

self.showtime\_combobox['values'] = showtimes

self.showtime\_combobox.current(0)

def generate\_ticket(self):

movie = self.movie\_combobox.get().strip()

showtime = self.showtime\_combobox.get().strip()

seat = self.seat\_spinbox.get().strip()

if not movie:

messagebox.showerror("Error", "Silakan pilih film!")

return

if not showtime:

messagebox.showerror("Error", "Silakan pilih jadwal tayang!")

return

if not seat.isdigit() or int(seat) <= 0 or int(seat) > 50:

messagebox.showerror("Error", "Silakan pilih kursi yang valid (1-50)!")

return

tanggal = datetime.now().strftime("%d %B %Y")

conn = mysql.connector.connect(

host="localhost",

user="root",

password="",

database="bioskop"

)

cursor = conn.cursor()

cursor.execute('SELECT studio FROM movies WHERE name = %s', (movie,))

studio = cursor.fetchone()[0]

conn.close()

ticket\_text = (

f"====================\n"

f" BIOSKOP 69 \n"

f"====================\n"

f"Film: {movie}\n"

f"Studio: {studio}\n"

f"Tanggal: {tanggal}\n"

f"Jadwal: {showtime}\n"

f"Kursi: {seat}\n"

f"====================\n"

f"Terima kasih telah memesan!\n"

f"===================="

)

self.ticket\_display.config(text=ticket\_text)

def update\_movie\_combobox(self):

conn = mysql.connector.connect(

host="localhost",

user="root",

password="",

database="bioskop"

)

cursor = conn.cursor()

cursor.execute('SELECT name FROM movies')

movies = cursor.fetchall()

conn.close()

movie\_names = [movie[0] for movie in movies]

self.movie\_combobox['values'] = movie\_names

if movie\_names:

self.movie\_combobox.current(0)

self.update\_showtimes(None)

def new\_method(self, conn):

cursor = conn.cursor()

return cursor # Memperbarui jadwal tayang berdasarkan film pertama

def add\_movie\_window(self):

# Jendela untuk menambah data film baru

add\_window = tk.Toplevel(self.root)

add\_window.title("Tambah Data Film")

name\_label = tk.Label(add\_window, text="Nama Film:")

name\_label.pack()

name\_entry = tk.Entry(add\_window)

name\_entry.pack()

studio\_label = tk.Label(add\_window, text="Studio:")

studio\_label.pack()

studio\_entry = tk.Entry(add\_window)

studio\_entry.pack()

showtimes\_label = tk.Label(add\_window, text="Jadwal Tayang (pisahkan dengan koma):")

showtimes\_label.pack()

showtimes\_entry = tk.Entry(add\_window)

showtimes\_entry.pack()

seats\_label = tk.Label(add\_window, text="Jumlah Kursi:")

seats\_label.pack()

seats\_entry = tk.Entry(add\_window)

seats\_entry.pack()

def save\_movie():

name = name\_entry.get()

studio = studio\_entry.get()

showtimes = showtimes\_entry.get()

seats = seats\_entry.get()

if name and studio and showtimes and seats.isdigit():

add\_movie(name, studio, showtimes, int(seats))

messagebox.showinfo("Sukses", "Film berhasil ditambahkan!")

add\_window.destroy()

self.update\_movie\_combobox() # Memperbarui combobox film setelah menambah data

else:

messagebox.showerror("Error", "Silakan isi semua data dengan benar!")

save\_button = tk.Button(add\_window, text="Simpan", command=save\_movie)

save\_button.pack(pady=10)

def edit\_movie\_window(self):

# Jendela untuk mengedit data film

edit\_window = tk.Toplevel(self.root)

edit\_window.title("Edit Data Film")

# Ambil daftar film dari database

conn = mysql.connector.connect(

host="localhost",

user="root",

password="",

database="bioskop"

)

cursor = conn.cursor()

cursor.execute('SELECT id, name FROM movies')

movies = cursor.fetchall()

conn.close()

if not movies:

messagebox.showinfo("Info", "Tidak ada film untuk diedit.")

return

movie\_names = [movie[1] for movie in movies]

movie\_combobox = ttk.Combobox(edit\_window, values=movie\_names)

movie\_combobox.pack()

name\_label = tk.Label(edit\_window, text="Nama Film Baru:")

name\_label.pack()

name\_entry = tk.Entry(edit\_window)

name\_entry.pack()

studio\_label = tk.Label(edit\_window, text="Studio Baru:")

studio\_label.pack()

studio\_entry = tk.Entry(edit\_window)

studio\_entry.pack()

showtimes\_label = tk.Label(edit\_window, text="Jadwal Tayang Baru (pisahkan dengan koma):")

showtimes\_label.pack()

showtimes\_entry = tk.Entry(edit\_window)

showtimes\_entry.pack()

seats\_label = tk.Label(edit\_window, text="Jumlah Kursi Baru:")

seats\_label.pack()

seats\_entry = tk.Entry(edit\_window)

seats\_entry.pack()

def save\_edit():

movie\_name = movie\_combobox.get()

movie\_id = next(movie[0] for movie in movies if movie[1] == movie\_name)

name = name\_entry.get()

studio = studio\_entry.get()

showtimes = showtimes\_entry.get()

seats = seats\_entry.get()

if name and studio and showtimes and seats.isdigit():

update\_movie(movie\_id, name, studio, showtimes, int(seats))

messagebox.showinfo("Sukses", "Data film berhasil diperbarui!")

edit\_window.destroy()

self.update\_movie\_combobox() # Memperbarui combobox film setelah mengedit data

else:

messagebox.showerror("Error", "Silakan isi semua data dengan benar!")

save\_button = tk.Button(edit\_window, text="Simpan Perubahan", command=save\_edit)

save\_button.pack(pady=10)

def delete\_movie\_window(self):

# Jendela untuk menghapus data film

delete\_window = tk.Toplevel(self.root)

delete\_window.title("Hapus Data Film")

# Ambil daftar film dari database

conn = mysql.connector.connect(

host="localhost",

user="root",

password="",

database="bioskop"

)

cursor = conn.cursor()

cursor.execute('SELECT id, name FROM movies')

movies = cursor.fetchall()

conn.close()

if not movies:

messagebox.showinfo("Info", "Tidak ada film untuk dihapus.")

return

movie\_names = [movie[1] for movie in movies]

movie\_combobox = ttk.Combobox(delete\_window, values=movie\_names)

movie\_combobox.pack()

def delete\_movie\_data():

movie\_name = movie\_combobox.get()

movie\_id = next(movie[0] for movie in movies if movie[1] == movie\_name)

delete\_movie(movie\_id)

messagebox.showinfo("Sukses", f"Film '{movie\_name}' berhasil dihapus!")

delete\_window.destroy()

self.update\_movie\_combobox() # Memperbarui combobox film setelah menghapus data

delete\_button = tk.Button(delete\_window, text="Hapus Film", command=delete\_movie\_data)

delete\_button.pack(pady=10)

if \_\_name\_\_ == '\_\_main\_\_':

root = tk.Tk()

app = TicketApp(root)

root.mainloop()

**BAB III**

**PENUTUP**

1. **KESIMPULAN**

Aplikasi ini merupakan program pemesanan tiket bioskop yang dirancang menggunakan Python, dengan integrasi database MySQL dan antarmuka grafis berbasis Tkinter. Program ini menyediakan solusi lengkap untuk mengelola data bioskop, mulai dari pembuatan database, pengelolaan data film (CRUD), hingga pemesanan tiket.

Fitur utama aplikasi meliputi:

1. Manajemen Database : Membuat dan mengelola tabel untuk menyimpan data film, jadwal tayang, studio, dan kursi.

2. GUI yang Intuitif : Menggunakan Tkinter untuk menyediakan antarmuka yang memudahkan pengguna dalam menambah, mengedit, menghapus data film, serta melakukan pemesanan tiket.

3. Pemesanan Tiket : Sistem memverifikasi pilihan pengguna untuk memastikan pemesanan berjalan lancar, mencetak tiket dalam format yang mudah dibaca.

Aplikasi ini menunjukkan bagaimana teknologi database dan Python dapat digunakan secara efisien untuk menciptakan solusi praktis dalam manajemen bioskop. Kehadiran GUI membuatnya lebih ramah pengguna dibandingkan dengan antarmuka berbasis teks tradisional.