ONLINE MOVIE TICKET BOOKING SYSTEM

DONE BY:

M.SANJAY 220701246

B.RAGUNANDAN 220701211

ABSTRACT

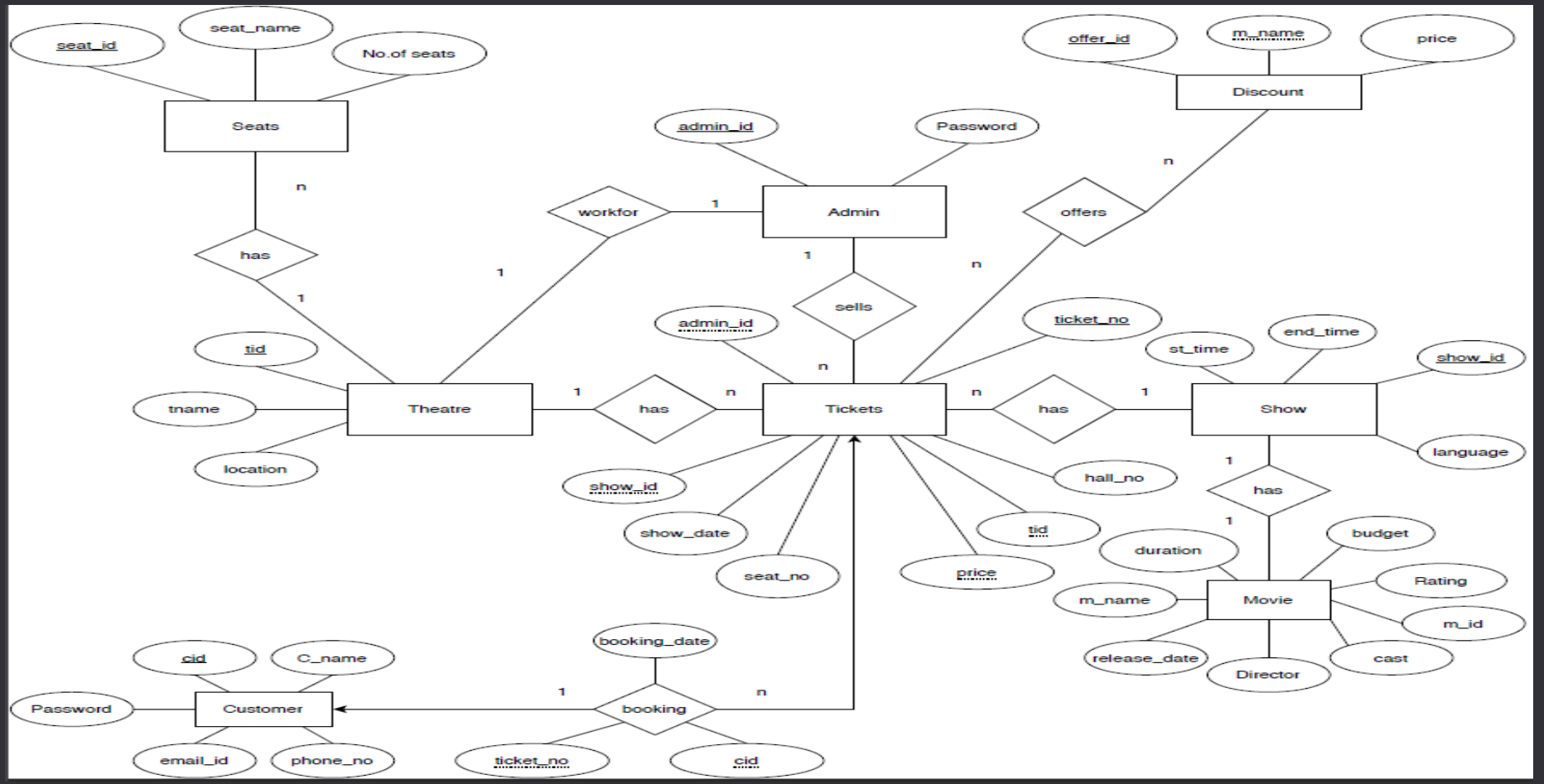
The online movie ticket booking system is a web-based application designed to facilitate the process of reserving tickets for movies in theaters. This system aims to enhance user convenience by allowing moviegoers to browse movie listings, select preferred showtimes, and purchase tickets from the comfort of their homes or while on the go. The system provides a user-friendly interface with features such as real-time seat selection, secure payment gateways, and instant confirmation of bookings. Additionally, it offers theaters an efficient way to manage ticket sales, track bookings, and reduce administrative overhead. By integrating features like promotional offers, user reviews, and movie trailers, the system also aims to enrich the user experience and drive engagement. This abstract outlines the key functionalities and benefits of the online movie ticket booking system, emphasizing its role in modernizing the ticket purchasing process and improving overall customer satisfaction.

KEY FETAURES

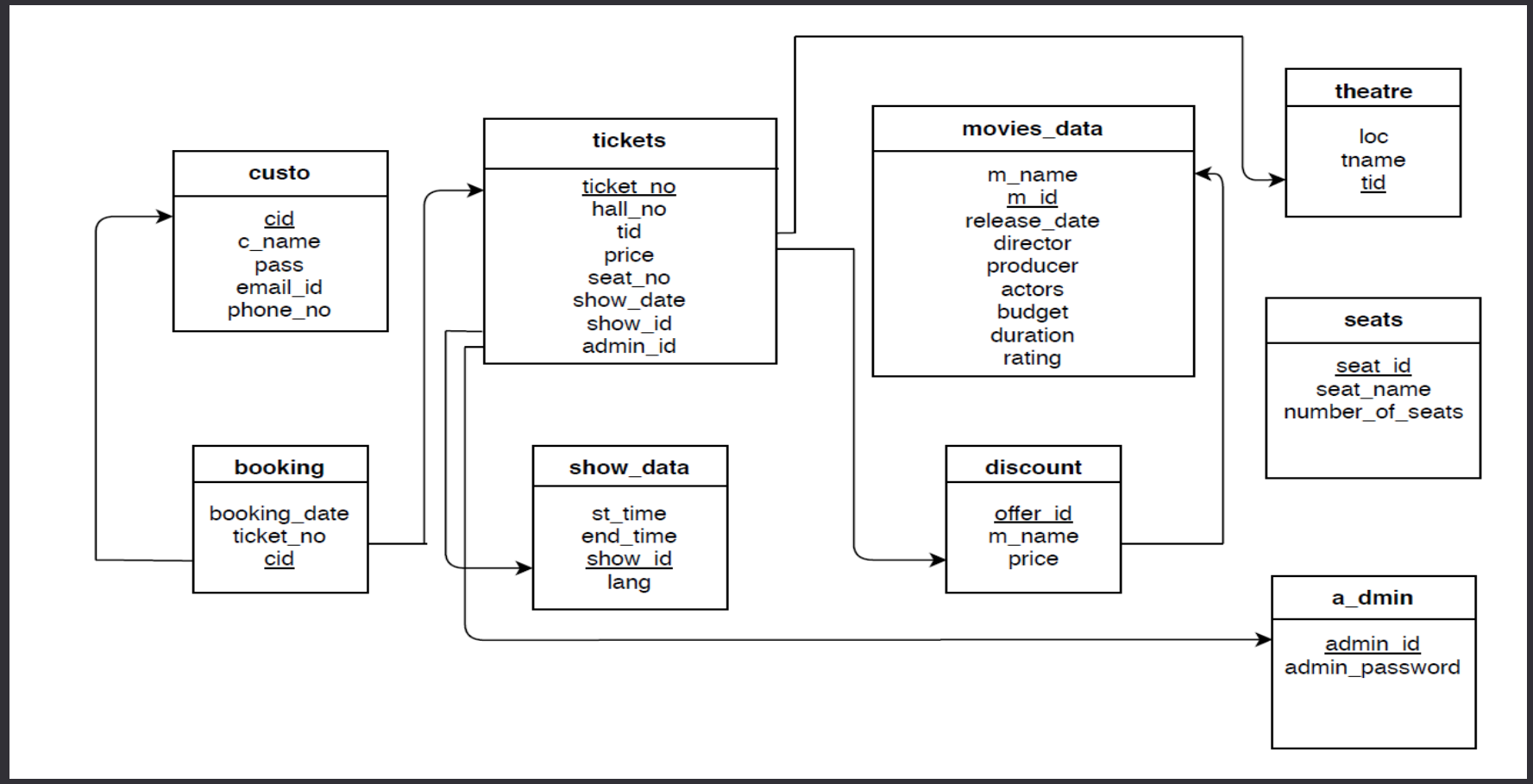
1.MOVIE INFO:
    1.MOVIE ID
    2.MOVIE NAME
    3.RELEASE DATE
    4.DIRECTOR
    5.CAST
    6.BUDGET
    7.DURATION
    8.RATING OF THE MOVIE

2.MOVIE DETAILS

ENTITY RELATIONSHIP DIAGRAM



SCHEMA DIAGRAM



BACKEND SOURCE CODE

```
Import sqlite3

def MovieData():
    con=sqlite3.connect("movie1.db")
    cur=con.cursor()
    cur.execute("CREATE TABLE IF NOT EXISTS book (id INTEGER PRIMARY KEY, Movie_ID text,Movie_Name text,Release_Date
text,Director text,Cast text,Budget text,Duration text,Rating text")
    con.commit()
    con.close()

def AddMovieRec(Movie_ID,Movie_Name,Release_Date,Director,Cast,Budget,Duration,Rating):
    con=sqlite3.connect("movie1.db")
    cur=con.cursor()
    cur.execute("INSERT INTO book VALUES (NULL, ?,?,?,?,?,?,?,?)",
(Movie_ID,Movie_Name,Release_Date,Director,Cast,Budget,Duration,Rating))
    con.commit()
    con.close()

def ViewMovieData():
    con=sqlite3.connect("movie1.db")
    cur=con.cursor()
    cur.execute("SELECT * FROM book")
    rows=cur.fetchall()
    con.close()
    return rows
```

```
def DeleteMovieRec(id):
    con=sqlite3.connect("movie1.db")
    cur=con.cursor()
    cur.execute("DELETE FROM book WHERE id=?", (id,))
    con.commit()
    con.close()

def SearchMovieData(Movie_ID="",Movie_Name="",Release_Date="",Director="",Cast="",Budget="",Duration="",Rating=""):
    con=sqlite3.connect("movie1.db")
    cur=con.cursor()
    cur.execute("SELECT * FROM book WHERE Movie_ID=? OR Movie_Name=? OR Release_Date=? OR Director=? OR Cast=? OR Budget=?
OR Duration=? OR Rating=?",(Movie_ID,Movie_Name,Release_Date,Director,Cast,Budget,Duration,Rating))
    rows=cur.fetchall()
    con.close()
    return rows

def UpdateMovieData(id,Movie_ID="",Movie_Name="",Release_Date="",Director="",Cast="",Budget="",Duration="",Rating=""):
    con=sqlite3.connect("movie1.db")
    cur=con.cursor()
    cur.execute("UPDATE book SET Movie_ID=?,Movie_Name=?,Release_Date=?,Director=?,Cast=?,Budget=?,Duration=?,Rating=?, WHERE
id=?",(Movie_ID,Movie_Name,Release_Date,Director,Cast,Budget,Duration,Rating))
    con.commit()
    con.close()
```

FRONTEND SOURCE CODE

```
from tkinter import *
import tkinter.messagebox
import MiniProject_Backend

class Movie:
        def __init__(self, root):
                self.root=root
                self.root.title("Online Movie Ticket Booking System")
                self.root.geometry("1350x750+0+0")
                self.root.config(bg="black")

                Movie_Name=StringVar()
                Movie_ID=StringVar()
                Release_Date=StringVar()
                Director=StringVar()
                Cast=StringVar()
                Budget=StringVar()
                Duration=StringVar()
                Rating=StringVar()

def iExit():
                        iExit=tkinter.messagebox.askyesno("Online Movie Ticket Booking System", "Are you sure???")
                        if iExit>0:
                                root.destroy()
                        return
```

```
def clcdata():
                                        self.txtMovie_ID.delete(0,END)
                                        self.txtMovie_Name.delete(0,END)
                                        self.txtRelease_Date.delete(0,END)
                                        self.txtDirector.delete(0,END)
                                        self.txtCast.delete(0,END)
                                        self.txtBudget.delete(0,END)
                                        self.txtRating.delete(0,END)
                                        self.txtDuration.delete(0,END)

                                def adddata():
                                        if(len(Movie_ID.get())!=0):

                        MiniProject_Backend.AddMovieRec(Movie_ID.get(),Movie_Name.get(),Release_Date.get(),Director.get(),Cast.get(),Budget.get(),D
uration.get(),Rating.get())
                                                        MovieList.delete(0,END)

                        MovieList.insert(END,(Movie_ID.get(),Movie_Name.get(),Release_Date.get(),Director.get(),Cast.get(),Budget.get(),Duration.get(),
Rating.get()))

def disdata():
                                        MovieList.delete(0,END)
                                        for row in MiniProject_Backend.ViewMovieData():
                                                MovieList.insert(END, row, str(""))

                                def movierec(event):
                                        global sd
                                        searchmovie=MovieList.curselection()[0]
                                        sd=MovieList.get(searchmovie)
                                        self.txtMovie_ID.delete(0,END)
                                        self.txtMovie_ID.insert(END,sd[1])
                                        self.txtMovie_Name.delete(0,END)
```

```
self.txtMovie_Name.insert(END,sd[2])
                                        self.txtRelease_Date.delete(0,END)
                                        self.txtRelease_Date.insert(END,sd[3])
                                        self.txtDirector.delete(0,END)
                                        self.txtDirector.insert(END,sd[4])
                                        self.txtCast.delete(0,END)
                                        self.txtCast.insert(END,sd[5])
                                        self.txtBudget.delete(0,END)
                                        self.txtBudget.insert(END,sd[6])
                                        self.txtDuration.delete(0,END)
                                        self.txtDuration.insert(END,sd[7])
                                        self.txtRating.delete(0,END)
                                        self.txtRating.insert(END,sd[8])

def deldata():
                                        if(len(Movie_ID.get())!=0):
                                                MiniProject_Backend.DeleteMovieRec(sd[0])
                                                clcdata()
                                                disdata()

                                def searchdb():
                                        MovieList.delete(0,END)
                                        for row in
MiniProject_Backend.SearchMovieData(Movie_ID.get(),Movie_Name.get(),Release_Date.get(),Director.get(),Cast.get(),Budget.get(),Duration.g
et(),Rating.get()):
                                                MovieList.insert(END, row, str(""))

                                def updata():
                                        if(len(Movie_ID.get())!=0):
                                                MiniProject_Backend.DeleteMovieRec(sd[0])
                                        if(len(Movie_ID.get())!=0):
```

```
MiniProject_Backend.AddMovieRec(Movie_ID.get(),Movie_Name.get(),Release_Date.get(),Director.get(),Cast.get(),Budget.get(),Duration.get(),Rating.get())
                                                MovieList.delete(0,END)

            MovieList.insert(END,(Movie_ID.get(),Movie_Name.get(),Release_Date.get(),Director.get(),Cast.get(),Budget.get(),Duration.get(),Rating.get()))

                                #Frames
                                MainFrame=Frame(self.root, bg="black")
                                MainFrame.grid()

                                TFrame=Frame(MainFrame, bd=5, padx=54, pady=8, bg="black", relief=RIDGE)
                                TFrame.pack(side=TOP)

                                self.TFrame=Label(TFrame, font=('Arial', 51, 'bold'), text="ONLINE MOVIE TICKET BOOKING SYSTEM", bg="black", fg="orange")
                                self.TFrame.grid()

                                BFrame=Frame(MainFrame, bd=2, width=1350, height=70, padx=18, pady=10, bg="black", relief=RIDGE)
                                BFrame.pack(side=BOTTOM)

                                DFrame=Frame(MainFrame, bd=2, width=1300, height=400, padx=20, pady=20, bg="black", relief=RIDGE)
                                DFrame.pack(side=BOTTOM)

                                DFrameL=LabelFrame(DFrame, bd=2, width=1000, height=600, padx=20, bg="black", relief=RIDGE, font=('Arial', 20, 'bold'), text="Movie Info_\n", fg="white")
                                DFrameL.pack(side=LEFT)

                                DFrameR=LabelFrame(DFrame, bd=2, width=450, height=300, padx=31, pady=3, bg="black", relief=RIDGE, font=('Arial', 20, 'bold'), text="Movie Details_\n", fg="white")
                                DFrameR.pack(side=RIGHT)
```

```
#Labels & Entry Box

                        self.lblMovie_ID=Label(DFrameL, font=('Arial', 18, 'bold'), text="Movie ID:", padx=2, pady=2, bg="black", fg="orange")
                        self.lblMovie_ID.grid(row=0, column=0, sticky=W)
                        self.txtMovie_ID=Entry(DFrameL, font=('Arial', 18, 'bold'), textvariable=Movie_ID, width=39, bg="black", fg="white")
                        self.txtMovie_ID.grid(row=0, column=1)

                        self.lblMovie_Name=Label(DFrameL, font=('Arial', 18, 'bold'), text="Movie Name:", padx=2, pady=2, bg="black",
fg="orange")
                        self.lblMovie_Name.grid(row=1, column=0, sticky=W)
                        self.txtMovie_Name=Entry(DFrameL, font=('Arial', 18, 'bold'), textvariable=Movie_Name, width=39, bg="black",
fg="white")
                        self.txtMovie_Name.grid(row=1, column=1)

                        self.lblRelease_Date=Label(DFrameL, font=('Arial', 18, 'bold'), text="Release Date:", padx=2, pady=2, bg="black",
fg="orange")
                        self.lblRelease_Date.grid(row=2, column=0, sticky=W)
                        self.txtRelease_Date=Entry(DFrameL, font=('Arial', 18, 'bold'), textvariable=Release_Date, width=39, bg="black",
fg="white")
                        self.txtRelease_Date.grid(row=2, column=1)
self.lblDirector=Label(DFrameL, font=('Arial', 18, 'bold'), text="Director:", padx=2, pady=2, bg="black", fg="orange")
                        self.lblDirector.grid(row=3, column=0, sticky=W)
                        self.txtDirector=Entry(DFrameL, font=('Arial', 18, 'bold'), textvariable=Director, width=39, bg="black", fg="white")
                        self.txtDirector.grid(row=3, column=1)

                        self.lblCast=Label(DFrameL, font=('Arial', 18, 'bold'), text="Cast:", padx=2, pady=2, bg="black", fg="orange")
                        self.lblCast.grid(row=4, column=0, sticky=W)
                        self.txtCast=Entry(DFrameL, font=('Arial', 18, 'bold'), textvariable=Cast, width=39, bg="black", fg="white")
                        self.txtCast.grid(row=4, column=1)
```

```
self.lblBudget=Label(DFrameL, font=('Arial', 18, 'bold'), text="Budget (Crores INR):", padx=2, pady=2, bg="black", fg="orange")
                        self.lblBudget.grid(row=5, column=0, sticky=W)
                        self.txtBudget=Entry(DFrameL, font=('Arial', 18, 'bold'), textvariable=Budget, width=39, bg="black", fg="white")
                        self.txtBudget.grid(row=5, column=1)

                        self.lblDuration=Label(DFrameL, font=('Arial', 18, 'bold'), text="Duration (Hrs):", padx=2, pady=2, bg="black",
fg="orange")
                        self.lblDuration.grid(row=6, column=0, sticky=W)
                        self.txtDuration=Entry(DFrameL, font=('Arial', 18, 'bold'), textvariable=Duration, width=39, bg="black", fg="white")
                        self.txtDuration.grid(row=6, column=1)

                        self.lblRating=Label(DFrameL, font=('Arial', 18, 'bold'), text="Rating (Out of 5):", padx=2, pady=2, bg="black",
fg="orange")
                        self.lblRating.grid(row=7, column=0, sticky=W)
                        self.txtRating=Entry(DFrameL, font=('Arial', 18, 'bold'), textvariable=Rating, width=39, bg="black", fg="white")
                        self.txtRating.grid(row=7, column=1)

#ListBox & ScrollBar
                        sb=Scrollbar(DFrameR)
                        sb.grid(row=0, column=1, sticky='ns')

                        MovieList=Listbox(DFrameR, width=41, height=16, font=('Arial', 12, 'bold'), bg="black", fg="white",
yscrollcommand=sb.set)
                        MovieList.bind('<<ListboxSelect>>', movierec)
                        MovieList.grid(row=0, column=0, padx=8)
                        sb.config(command=MovieList.yview)

                        #Buttons
                        self.btnadd=Button(BFrame, text="Add New", font=('Arial', 20, 'bold'), width=10, height=1, bd=4, bg="orange",
command=adddata)
                        self.btnadd.grid(row=0, column=0)
```

```
self.btndis=Button(BFrame, text="Display", font=('Arial', 20, 'bold'), width=10, height=1, bd=4, bg="orange", command=disdata)
                        self.btndis.grid(row=0, column=1)

                        self.btnclc=Button(BFrame, text="Clear", font=('Arial', 20, 'bold'), width=10, height=1, bd=4, bg="orange",
command=clcdata)
                        self.btnclc.grid(row=0, column=2)

                        self.btnse=Button(BFrame, text="Search", font=('Arial', 20, 'bold'), width=10, height=1, bd=4, bg="orange",
command=searchdb)
                        self.btnse.grid(row=0, column=3)

                        self.btndel=Button(BFrame, text="Delete", font=('Arial', 20, 'bold'), width=10, height=1, bd=4, bg="orange",
command=deldata)
                        self.btndel.grid(row=0, column=4)

                        self.btnup=Button(BFrame, text="Update", font=('Arial', 20, 'bold'), width=10, height=1, bd=4, bg="orange",
command=updata)
                        self.btnup.grid(row=0, column=5)

                        self.btnx=Button(BFrame, text="Exit", font=('Arial', 20, 'bold'), width=10, height=1, bd=4, bg="orange",
command=iExit)
                        self.btnx.grid(row=0, column=6)


if __name__=='__main__':
            root=Tk()
            datbase=Movie(root)
            root.mainloop()
```

THANK YOU