Computer Project 22-23

Theatre Ticket Booking System

By:
Aadarsh Hari
Harish Chandar.C
Sarvesh.P
Rohit Sundar.V

Table Of Contents

About The Project

The aim of the project is to create an object-oriented program to book a ticket for a movie in a theater.

Objectives:

- *Create Account:*
  - *We ask the user to create a new account with their Email ID and password and store it a CSV file.*

- *Main Page:*
  - *We get name, age and the number of seats and confirmation for premium account from the user and store the information into a CSV file.*

- *Selecting Movie:*
  - *A plethora of movies are on display and the user can chose the one for their liking by clicking on the movie's name.*

➢ *Selecting seats:*

- *The user can choose the seats to their liking and the number of seats chosen is up to the user. All these information is stored in a CSV file.*

➢ *Payment:*

- *A bill with the required amount is shown as a pop up to the user with a redirect button which will redirect them to the payment page.*

- *After payment, a pop-up message will show saying that the transaction was successful and a receipt will be sent to the user's Email ID.*

Hardware and Software required:

➢ *Software:*

▪ *Operating System:*

- *Windows 7 or above, MacOS High Sierra*

▪ *Front End:*

- Python 3.8 or above (preferably 3.10)

- *Back End:*
  - Python CSV Module

➢ *Hardware:*

- *Processor:* Intel i3 or better / AMD Ryzen 3 or better
- *RAM:* 4 GB or more
- *Storage:* 1 GB of free space
- Stable internet connection

About Python:

Python is a high-level, general-purpose programming language. One of its strong points is code readability, and dynamic typing. It supports

structured as well as object-oriented programming. The various modules available for users increase the scope of vanilla python a housand-fold. This in pair with its ease of use for beginners, as well as its various tools (modules) on offer to pioneers, make it a leading programming language in today's world.

Modules Used:

- *TKinter: This is used to create the frontend of the entire project, and improve user experience.*

- *Python Imaging Library (PIL): The Python Imaging Library adds image processing capabilities to your Python interpreter. The library provides extensive file format support, an efficient internal representation and fairly powerful image processing capabilities.*

- *CSV (Comma Separated Values): This enables the usage of a CSV file to obtain and append the Name, Age, No. of seats,*

Confirmation of Premium, Movie name, Movie Language, Selected seats, Cast in a CSV file.

Structure of the Project:

- *The code contains 23 functions in total, 14 in frontend.py, 4 in backend.py and 5 in user_ management.py*

Front End:

- *def user_input( ): We are creating an instance of a GUI window using TKinter and we are getting the required input.*
  - *def create( ): This is a sub-function of the above-mentioned function. The data is appended in a CSV file using backend.py. It checks if the required spaces are empty or not.*
- *def book_ticket( ): We get required input from the user and book a ticket.*

- *def submit( ): Error is checked from the gotten input and if error is found, a pop-up message is shown to the user and if there is no error found, the data is appended into data.csv file and the TKinter window is destroyed.*

- *def select_movie( ): A page with a plethora of movies airing at the time is introduced to the user with their names and posters.*

    - *def movie( ): When a movie is selected, the pre-defined details of the movie are appended into data.csv and the TKinter window which was created in the above function is destroyed*

- *def select_seating( ): A new Tkinter window is created and the page for selecting the desired seats is introduced to the user.*

- *def buttons( ): A button is assigned for each seat using codes for coordination of the placement of seats.*

- *def button_clicked(seat):*

  The seats are taken as a string and it's appended to the empty list "selected_seats".

- *def submit_seat( ):The above-mentioned list is written into data.csv*

- *def payment( ): A new TKinter window is opened and the required amount is shown to the user with a button to redirect the user to the next stage of payment.*

- *def thank_you( ):A Tkinter window is created which shows the user about the status of payment and sends a message about the receipt of the payment.*

- *def ok_command( ):All the opened TKinter windows are destroyed.*

Back End:

- *def input_data(name,age,set_nos,premium): CSV module is imported and all the values of the parameters are appended into data.csv*

- *def input_userData(gmail,password): The parameters are gotten from the user and is then written on user_data.csv.*

- *def write_newData(movie,genre,movie_lang,cast): The data of the parameters are gotten from the user and is appended into the file data.csv.*

- *def write_seatData(seats): The parameter is entered into an empty list and is appended into data.csv similar to the previous functions*

User Management:

- *def delete: Desired input is gotten from the user and the matching record is deleted from the csv file.*

- *def sort_update( ): Desired input is gotten from the user and the matching record is updated in the csv file.*

- *def update( ): Desired input is gotten form the user and any change or error is corrected by using this function.*

Algorithm

- *Front End:*
  - *User Input:*
    - *Importing necessary modules (TKinter, PIL, backend)*
    - *Defining an instance of TKinter.*
    - *Declare the dimension of the window and creating it.*
    - *Create a new label "Create new Account" and placing it on the window.*
    - *Creating another label "Enter Gmail" and placing it on the window using coordinates.*
    - *Creating a label "Enter Password" and placing it on the window using coordinates.*

- *Declaring an entry for Gmail and placing it on the window.*
- *Declaring another entry for password and placing it on the window.*
- *Creating a button "Create Account" and placing it on the window using coordinates with the command "create".*

❖ *Create:*

- *Getting Email and password from user.*
- *If any of those fields are empty, an error is thrown or else the window is destroyed.*

❖ *Book Ticket:*

- *Creating a new TKinter instance.*
- *Creating a new TKinter window with specific dimensions, icon and title.*

- *Defining a new image using PIL and the selected image depicts a movie theatre.*
- *Creating a label "Book your Show" and placing it on the window using coordinates.*
- *Creating a label "Enter Name" and placing it on the window using coordinates.*
- *Creating a label "Enter Number of Seats" and placing in on the window using coordinates.*
- *Creating an entry for each of these fields.*
- *Creating a radiobutton for validity of "Premium".*
- *Creating a button "Submit" with is command as "submit".*

- *Creating another button "Quit" with its command as to destroy the window.*

❖ *Submit:*

- *Above mentioned input is received from the user and is checked whether is the field is empty.*
- *If any said field is empty, an error is thrown mentioning the field which is empty or invalid.*
- *Or else, if there are no errors, the data is appended to "data.csv". and the window is destroyed.*

❖ *Select Movie:*

- *Creating a new TKinter instance.*
- *Creating a new TKinter window with specific dimensions, icon and title.*

- *Defining a new image using PIL and the selected image depicts a movie theatre.*

- *A plethora of movies are displayed for the user to choose from. Corresponding buttons are placed for the user to select the desired movie.*

- *When a movie is selected, pre-defined data such as movie name, genre, cast, movie language, are appended into the file "data.csv" and the window is destroyed.*

❖ *Select Seating:*

- *Creating a new TKinter instance.*

- *Creating a new TKinter window with specific dimensions, icon and title.*

- *Defining a new image using PIL and the selected image depicts a screen seating.*

- *We are creating a new canvas and we are placing buttons corresponding to the seats.*

- *We are placing another button "Submit" with its command as "submit_seats".*

❖ *Button Clicked:*

- *Selected seats are appended into a list which holds all the information of the selected seats.*

❖ *Submit Seats:*

- *Above mentioned list is appended into "data.csv".*

❖ *Payment:*

- *A new TKinter window is displayed showing the payment status and required amount.*

- ❖ *Thank You:*
  - *A new window is created to thank the user.*

- ➢ *Back End:*
  - ❖ *Input Data:*
    - *Inputted details of the user are taken as parameters and is appended into "data.csv".*
  - ❖ *Input User Data:*
    - *Gmail and password are taken as inputs and are appended into "user_data.csv".*
  - ❖ *Write New Data:*
    - *"Movie", "Movie Language", "Cast", "Genre" are taken as parameters.*
    - *The existing data of "data.csv" is appended to a list "data".*

- *New file object is created. Above mentioned parameters and the existing data is appended into the same file "data.csv" and closed.*

❖ *Write Seat Data:*

- *Selected seats are taken as parameters.*
- *Existing Data of "data.csv" is appended into a list "data".*
- *We are appending selected seats into "data" and rewriting it in the same file and closing it.*

➢ *User Management:*

❖ *Delete Record:*

- *We are opening "data.csv" and getting desired name as input.*

- *If desired name matches any name in "data.csv", said record is deleted.*

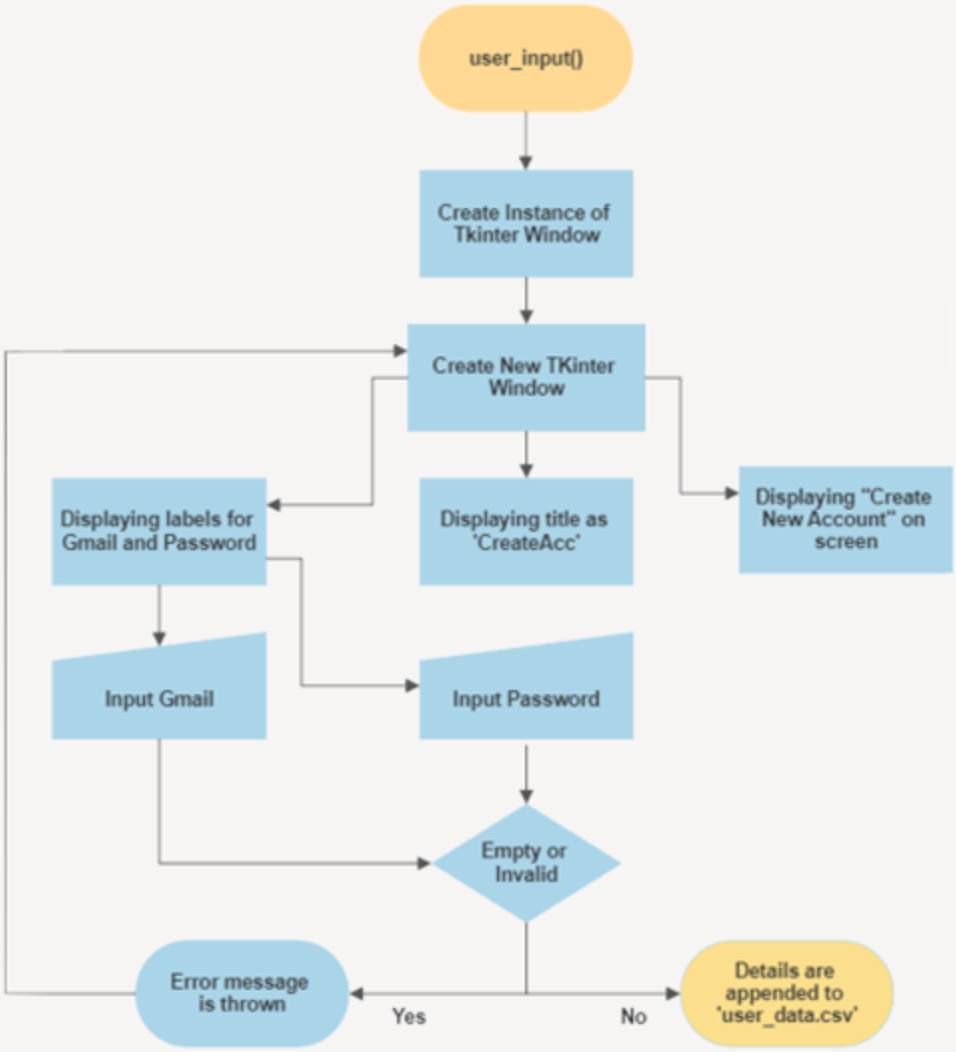
❖ *Sort and Update Records:*

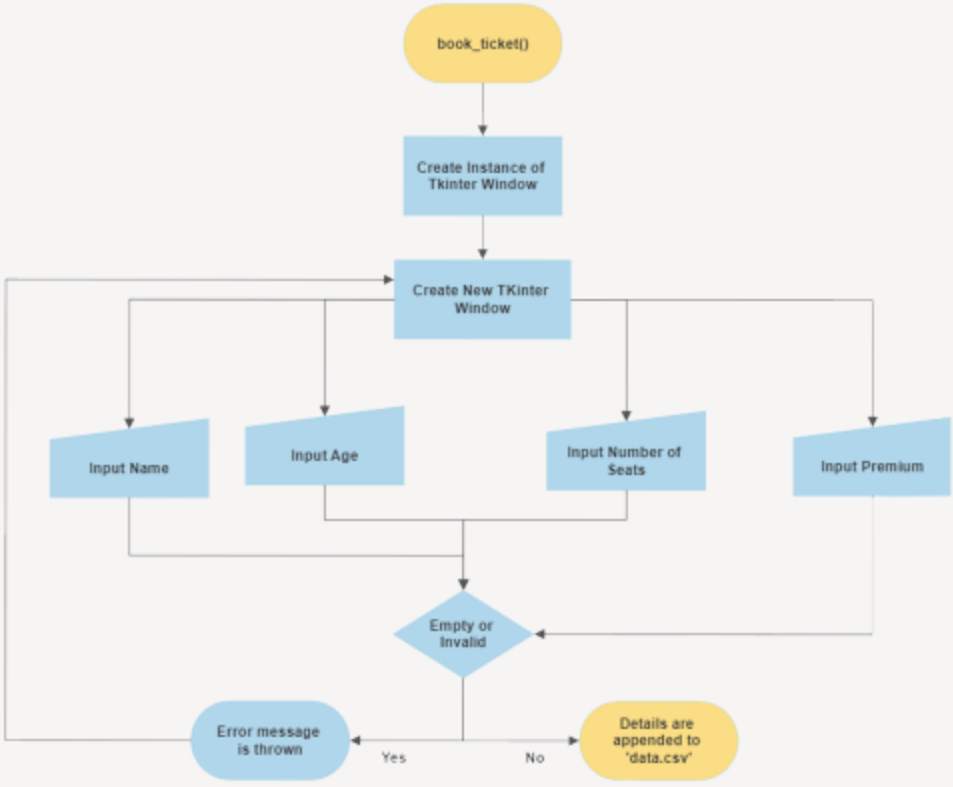
- *Desired parameter is gotten from user and "data.csv" is sorted accordingly.*

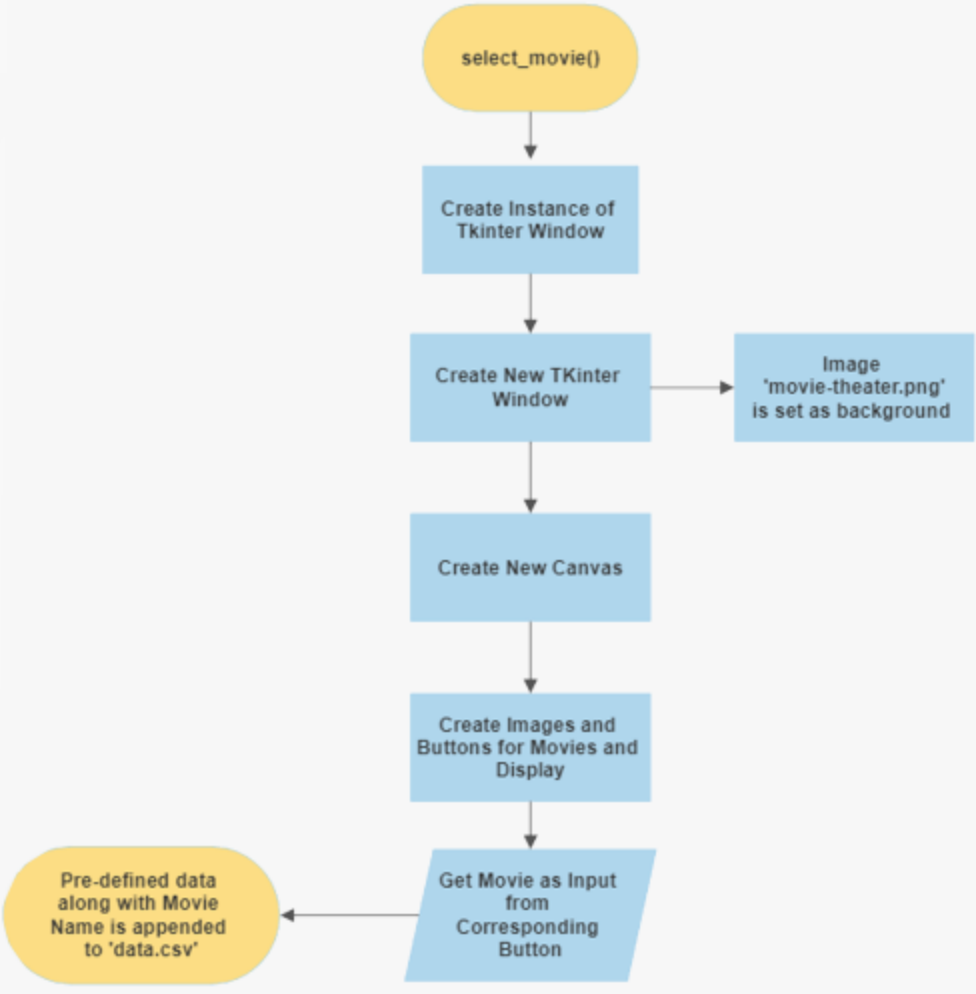
❖ *Update Specific Records:*

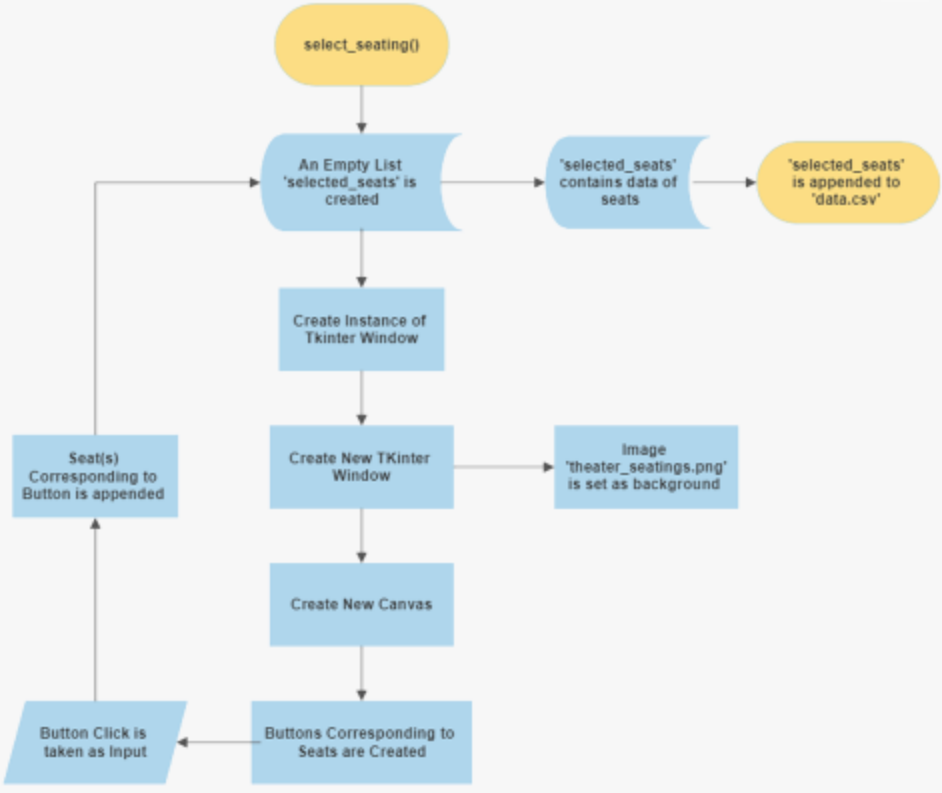
- *Desired name and field are taken as input from the user and the matching record in "data.csv" is updated.*

FLOWCHART


user_input()
Create Instance of Tkinter Window
Create New TKinter Window
Displaying labels for Gmail and Password
Displaying title as 'CreateAcc'
Displaying "Create New Account" on screen
Input Gmail
Input Password
Empty or Invalid
Error message is thrown
Yes
No
Details are appended to 'user_data.csv'


book_ticket()
Create Instance of Tkinter Window
Create New TKinter Window
Input Name
Input Age
Input Number of Seats
Input Premium
Empty or Invalid
Error message is thrown
Yes
No
Details are appended to 'data.csv'


select_movie()
Create Instance of Tkinter Window
Create New TKinter Window
Image 'movie-theater.png' is set as background
Create New Canvas
Create Images and Buttons for Movies and Display
Get Movie as Input from Corresponding Button
Pre-defined data along with Movie Name is appended to 'data.csv'


select_seating()
An Empty List 'selected_seats' is created
'selected_seats' contains data of seats
'selected_seats' is appended to "data.csv"
Create Instance of Tkinter Window
Create New TKinter Window
Image "theater_seatings.png" is set as background
Create New Canvas
Buttons Corresponding to Seats are Created
Button Click is taken as Input
Seat(s) Corresponding to Button is appended


input_data()
"name","age","seat_no",
"premium" are
paramenters
Open "data.csv " in "a"
mode
Create writer object
Using .writerow()
append data to csv file
Close file


input_userData()
"gmail","password" are parameters
Open "user_data.csv" in "a" mode
Create writer object
Using .writerow() append data to csv file
Close file


write_newData()
"movie","genre","movie_lang", "cast"are parmeters
Open "data.csv" in "r" mode
Store old data in empty list
Add new data to list
Reopen "data.csv" in "w" mode
Using .writerow() append new data to csv file
Close file


del_rec()
Open "data.csv " in "r" mode
Create writer object
According to user input store required data in empty list using for loop and conditional statements
Reopen "data.csv" in "w" mode
Using .writerow() append new data to csv file
Close file

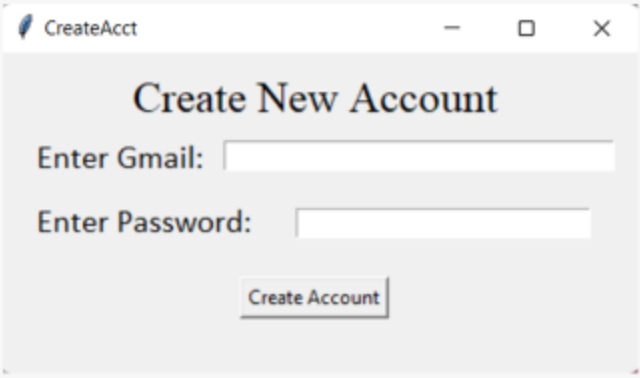

sort_lang()
Open "data.csv " in "r" mode
Create writer object
Using for loops and if statements perform sorting
Store sorted data in empty list
Reopen "data.csv" in "w" mode
Using .writerow() append new data to csv file
Close file

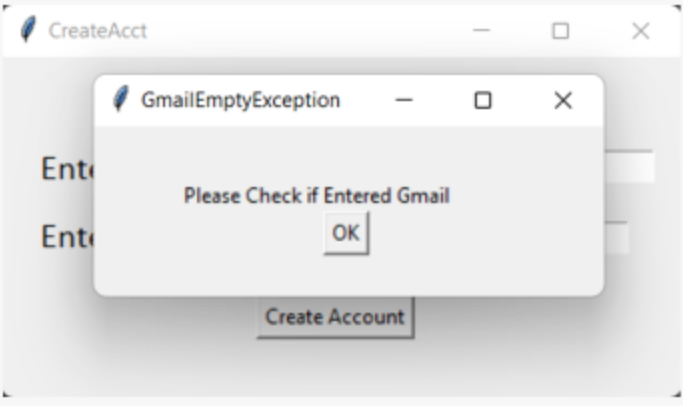

sort_genre()
Open "data.csv " in "r" mode
Create writer object
Using for loops and if statements perform sorting
Store sorted data in empty list
Reopen "data.csv" in "w" mode
Using .writerow() append new data to csv file
Close file

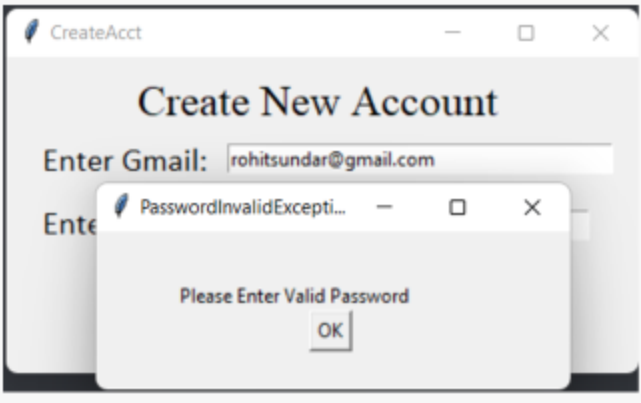

sort_cast()
Open "data.csv " in "r" mode
Create writer object
Using for loops and if statements perform sorting
Store sorted data in empty list
Reopen "data.csv" in "w" mode
Using .writerow() append new data to csv file
Close file

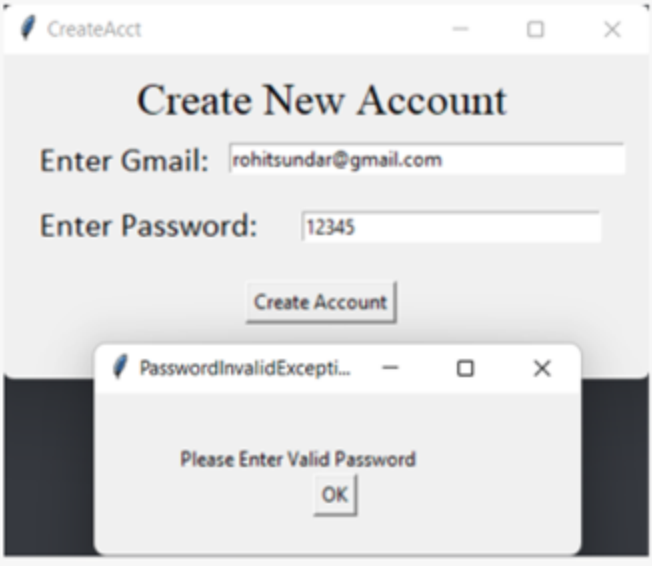

sort_cast()
Open "data.csv " in "r" mode
Create writer object
Using for loops and if statements perform sorting
Store sorted data in empty list
Reopen "data.csv" in "w" mode
Using .writerow() append new data to csv file
Close file

OUTPUT


CreateAcct
Create New Account
Enter Gmail:
Enter Password:
Create Account


CreateAcct
GmailEmptyException
Please Check if Entered Gmail
OK
Create Account


CreateAcct
Create New Account
Enter Gmail:
rohitsundar@gmail.com
PasswordInvalidExcepti...
Please Enter Valid Password
OK

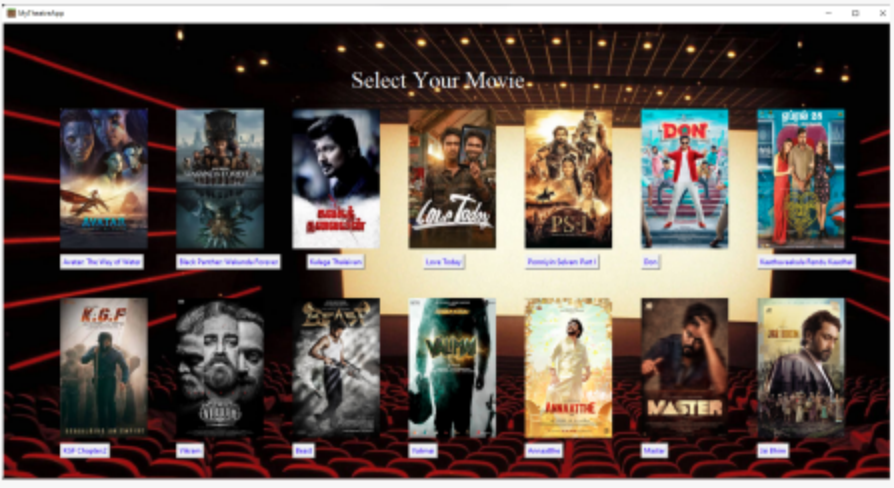

CreateAcct
Create New Account
Enter Gmail:
rohitsundar@gmail.com
Enter Password:
12345
Create Account
PasswordInvalidExcepti...
Please Enter Valid Password
OK

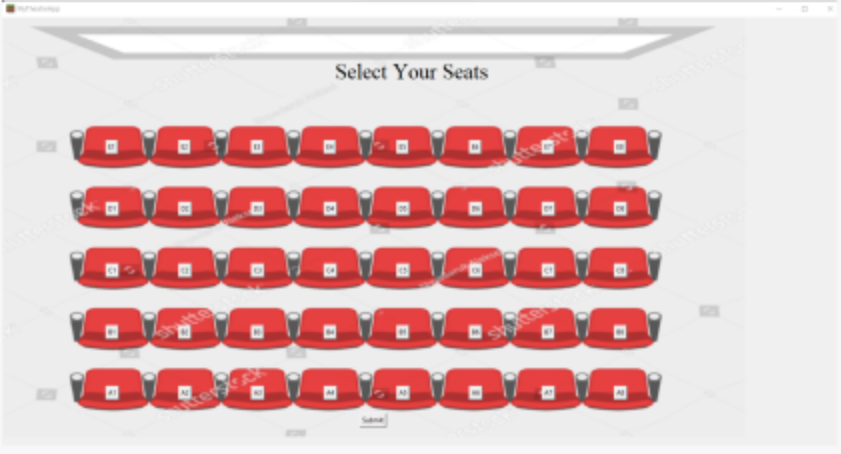

Book Your Show

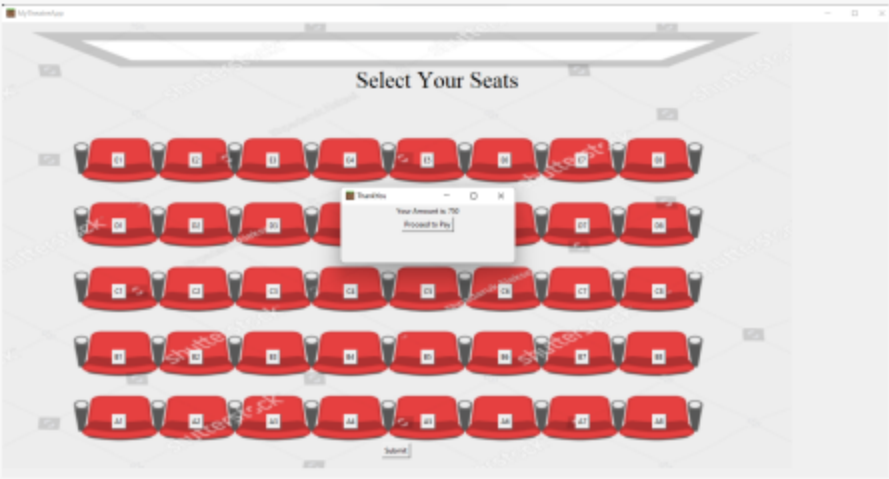

Book Your Show
Enter Name:
Rohit Sundar
Enter Age:
Enter Number of Seats:
AgeEmptyException
Please Check if Entered Field Age
OK
Select Premiums
YES
NO
Submit
Quit

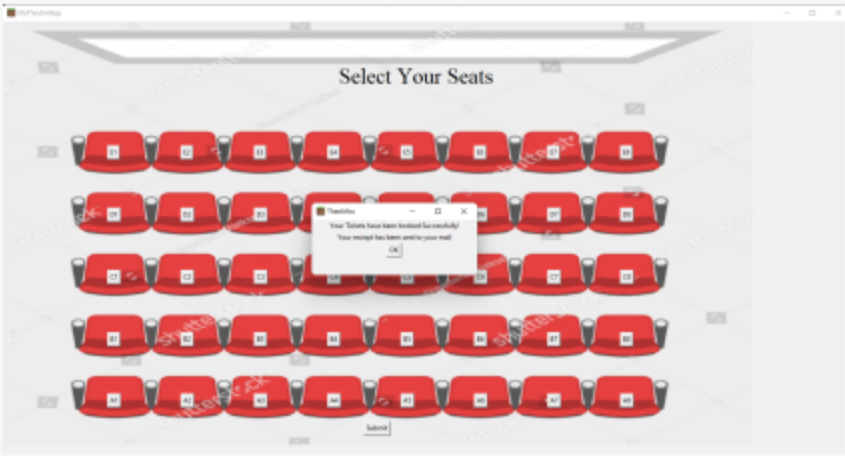

Book Your Show
Enter Name:
Rohit Sundar
Enter Age:
Enter Number of Seats:
AgeEmptyException
Please Check if Entered Field Age
OK
Select Premiums
YES
NO
Submit
Quit


Book Your Show


Select Your Movie
Love Today
PS-1
DON
K.G.F
VIKRAM
VAATHI
VAARISU
VAATHI
MASTER


Select Your Seats
Submit


Select Your Seats
Proceed to Pay
Submit


Select Your Seats
OK
Submit

SOURCE CODE
FRONT END

```
# Theatre Ticket Booking System
# Done by: Aadarsh of 12-A, Mohit of 12-A, Harish Chandar of 12-C, Harvesh of 12-C

# Importing necessary module(s)
from tkinter import *
from PIL import ImageTk, Image
import backend


# Defining the Frontend of the Customer
def frontend():


    def user_input():
        # Defining an instance of a GUI window using TKinter
        child1 = Tk()
        child1.title("CreateAcct")
        child1.geometry('400x200')

        # Labels are the text that appears on the GUI window
        # Entries are the places where we input the data
        """ .place() is a method associated with TKinter to display the object (here: Label, Entry or Radiobutton)
            on the GUI using coordinates"""

        head_label = Label(child1, text = "Create New Account", font = ("Times New Roman", 20)).place(x=80, y=10)

        gmail_label = Label(child1, text = "Enter Gmail: ", font = ("Calibri", 15)).place(x = 20, y = 50)
        gmail_entry = Entry(child1, width = 40)
        gmail_entry.place(x = 140, y = 55)

        pass_label = Label(child1, text="Enter Password: ", font=("Calibri", 15)).place(x = 20, y = 90)
        pass_entry = Entry(child1, width=30)
        pass_entry.place(x=185, y=97)

        """This function will be called only when the button "create_button"
        (defined below) is clicked on, or when the function is called"""


            def create():
                # .get() method is associated with TKinter to get the value from the entry field
                gmail = gmail_entry.get()
                password = pass_entry.get()

                backend.input_userData(gmail, password)
                if gmail == "":

                    child1_1 = Tk()
                    child1_1.title("GmailEmptyException")
                    child1_1.geometry("300x100")

                    gmail_er = Label(child1_1, text="Please Check if Entered Gmail").place(x=50, y=30)
                    ok_button = Button(child1_1, text="OK", command=child1_1.destroy).place(x=135, y=50)

                elif password == "" or len(password) < 6:

                    child1_2 = Tk()
                    child1_2.title("PasswordInvalidException")
                    child1_2.geometry("300x100")

                    gmail_er = Label(child1_2, text="Please Enter Valid Password").place(x=50, y=30)
                    ok_button = Button(child1_2, text="OK", command=child1_2.destroy).place(x=135, y=50)

                    child1_2.mainloop()

                else:

                    child1.destroy()


            create_button = Button(child1, text="Create Account", command=create)
            create_button.place(x=150, y=140)

            child1.mainloop()

        user_input()
```

```
def book_ticket():

    parent1  = Tk()
    parent1.title("MyTheatreApp")
    parent1.geometry('1920x1080')
    parent1.iconbitmap(r'icon/minecraft-icon-24.ico')

    bg = ImageTk.PhotoImage(Image.open(r".\theatre\movie-theater.png"))
    bg_label = Label(image=bg).pack()

    """Radiobuttons are where we select and highlight the predefined choices available for the user.
    It is just the same "circle" buttons in a Google Form"""

    head_label = Label(parent1, text ="Book Your Show", font = ("Times New Roman", 30)).place(x = 610, y = 50)

    name_label = Label(parent1, text="Enter Name: ").place(x=510, y=150)
    name_entry = Entry(parent1, width=40)
    name_entry.place(x=710, y=150)

    age_label = Label(parent1, text ="Enter Age: ").place(x = 510, y = 250)
    age_entry = Entry(parent1, width = 10)
    age_entry.place(x = 710, y = 250 )

    seat_nos_label = Label(parent1, text ="Enter Number of Seats: ").place(x = 510, y = 350)
    seat_nos_entry = Entry(parent1, width = 10)
    seat_nos_entry.place(x = 710, y = 350)

    #Assigning a Boolean Variable to "prem" to store a true or false value
    prem = BooleanVar()

    prem_label = Label(parent1, text ="Select Premium: ").place(x = 510, y = 450)
    prem_radbtn_true = Radiobutton(parent1, text ="YES", variable = prem, value = True).place(x = 710, y = 450)
    prem_radbtn_false = Radiobutton(parent1, text ="NO", variable = prem, value = False).place(x = 860, y = 450)




    def submit():
        name = name_entry.get()
        age = age_entry.get()
        seat_nos = seat_nos_entry.get()
        premium = prem.get()

        """ Some conditions to check whether an entry field is empty and open a new child instance
        of a GUI window to show error messages"""

        if name == "":
            tk1 = Tk()
            tk1.title("NameEmptyException")
            tk1.geometry("300x100")

            name_er = Label(tk1, text = "Please Check if Entered Name").place(x = 50, y = 30)
            ok_button = Button(tk1, text = "OK", command = tk1.destroy).place(x = 135, y = 50)

            tk1.mainloop()

        elif age == "":
            tk2 = Tk()
            tk2.title("AgeEmptyException")
            tk2.geometry("300x100")

            age_er = Label(tk2, text = "Please Check if Entered Valid Age").place(x = 50, y = 30)
            ok_button = Button(tk2, text = "OK", command = tk2.destroy).place(x = 135, y = 50)

            tk2.mainloop()

        elif seat_nos == "":
            tk3 = Tk()
            tk3.title("SeatsEmptyException")
            tk3.geometry("300x100")

            seat_nos_er = Label(tk3, text = "Please Check if Entered Valid Number Of Seats").place(x = 50, y = 30)
            ok_button = Button(tk3, text = "OK", command = tk3.destroy).place(x = 135, y = 50)

            tk3.mainloop()
```

```
    else:
        ''' "backend" is another python file which we have imported earlier which is in
        the same directory as this program. Here ".input_data" is a method associated with "backend" but it is a
        function in that file. To summarise, we are just calling a function which is in another python file'''

        backend.input_data(name, age, seat_nos, premium)
        parent1.destroy()
        select_movie()


submit_button = Button(parent1, text ="Submit", command = submit)
submit_button.place(x = 590, y = 600)

quit_button = Button(parent1, text ="Quit", command = parent1.destroy)
quit_button.place(x = 730, y = 600)


def select_movie():

    parent2  = Tk()
    parent2.title("MyTheatreApp")
    parent2.geometry("1920x1080")
    parent2.iconbitmap(r".\icon\minecraft-icon-24.ico")

    movie_theater = ImageTk.PhotoImage(Image.open(r".\theatre\movie-theater.png"))

    my_canvas = Canvas(parent2, width=1920, height=1080)
    my_canvas.pack(fill="both", expand=True)
    my_canvas.create_image(0, 0, image=movie_theater, anchor="nw")
    my_canvas.create_text(750, 100, text="Select Your Movie", font=("Times New Roman", 30), fill="White")




        def movie1():

            parent2.destroy()
            movie = "Avatar: The Way of Water"
            genre = "Fantasy"
            movie_language = "English"
            cast = ("Sam Worthington", "Zoey Saldana")
            select_seating()

            backend.write_newData(movie, genre, movie_language, cast)


        def movie2():

            parent2.destroy()
            movie = "Black Panther: Wakanda Forever"
            genre = "Superhero"
            movie_language = "English"
            cast = ("Chadwick Boseman","Letitia Wright")
            select_seating()

            backend.write_newData(movie, genre, movie_language, cast)


        def movie3():

            parent2.destroy()
            movie = "Kalaga Thalaivan"
            genre = "Action"
            movie_language = "Tamil"
            cast = ("Udhayanidhi Stalin","Nidhhi Agarwal")
            select_seating()

            backend.write_newData(movie, genre, movie_language, cast)
```

```
def movie4():

    parent2.destroy()
    movie = "Love Today"
    genre = "Comedy"
    movie_language = "Tamil"
    cast = ("Pradeep Ranganadhan","Nikita")
    select_seating()

    backend.write_newData(movie, genre, movie_language, cast)


def movie5():

    parent2.destroy()
    movie = "Ponniyin Selvan: Part I"
    genre = "Historical Drama"
    movie_language = "Tamil"
    cast = ("Jayam Ravi","Karthi","Vikram",
    "Aishwarya Rai","Trisha Krishnan")
    select_seating()


    backend.write_newData(movie, genre, movie_language, cast)


def movie6():

    parent2.destroy()
    movie = "Don"
    genre = "Romance"
    movie_language = "Tamil"
    cast = ("Sivakarthikeyan","Priyanka Mohan")
    select_seating()

    backend.write_newData(movie, genre, movie_language, cast)



        def movie7():

            parent2.destroy()
            movie = "Kaathuvaakula Rendu Kaadhal"
            genre = "Romantic Comedy"
            movie_language = "Tamil"
            cast = ("Vijay Sethupathi","Nayanthara", "Samantha Ruth Prabhu")
            select_seating()

            backend.write_newData(movie, genre, movie_language, cast)


        def movie8():

            parent2.destroy()
            movie = "KGF Chapter:2"
            genre = "Historical Drama"
            movie_language = "Tamil"
            cast = ("Yash", "Srinidhi Shetty")
            select_seating()

            backend.write_newData(movie, genre, movie_language, cast)


        def movie9():

            parent2.destroy()
            movie = "Vikram"
            genre = "Action Thriller"
            movie_language = "Tamil"
            cast = ("Kamal Haasan", "Vijay Sethupathi")
            select_seating()

            backend.write_newData(movie, genre, movie_language, cast)
```

```
def movie10():

    parent2.destroy()
    movie = "Beast"
    genre = "Thriller"
    movie_language = "Tamil"
    cast = ("Vijay", "Pooja Hegde")
    select_seating()

    backend.write_newData(movie, genre, movie_language, cast)


def movie11():

    parent2.destroy()
    movie = "Valimai"
    genre = "Action"
    movie_language = "Tamil"
    cast = ("Ajith Kumar","Huma Qureshi")
    select_seating()

    backend.write_newData(movie, genre, movie_language, cast)


def movie12():

    parent2.destroy()
    movie = "Annaatthe"
    genre = "Drama"
    movie_language = "Tamil"
    cast = ("Rajinikanth", "Keerthy Suresh")
    select_seating()

    backend.write_newData(movie, genre, movie_language, cast)
```

```
def movie13():

    parent2.destroy()
    movie = "Master"
    genre = "Mystery"
    movie_language = "Tamil"
    cast = ("Vijay","Vijay Sethupathi")
    select_seating()

    backend.write_newData(movie, genre, movie_language, cast)


def movie14():
    parent2.destroy()
    movie = "Jai Bhim"
    genre = "Legal Drama"
    movie_language = "Tamil"
    cast = ("Suriya", "Lijmol Jose")
    select_seating()

    backend.write_newData(movie, genre, movie_language, cast)


movie1_poster = ImageTk.PhotoImage(Image.open(r".\movies\avatar_the_way_of_water.png"))
my_canvas.create_image(100, 150, image=movie1_poster, anchor="nw")
movie1_button = Button(parent2, text="Avatar: The Way of Water", fg='Blue', command=movie1)
movie1_buttonWindow = my_canvas.create_window(100, 400, anchor="nw", window=movie1_button)
```

```
movie2_poster = ImageTk.PhotoImage(Image.open(r".\movies\black_panther_wakanda_forever.png"))
my_canvas.create_image(300, 150, image=movie2_poster, anchor="nw")
movie2_button = Button(parent2, text="Black Panther: Wakanda Forever", fg='Blue', command=movie2)
movie2_buttonWindow = my_canvas.create_window(300, 400, anchor="nw", window=movie2_button)


movie3_poster = ImageTk.PhotoImage(Image.open(r".\movies\kalaga_thalaivan.png"))
my_canvas.create_image(500, 150, image=movie3_poster, anchor="nw")
movie3_button = Button(parent2, text="Kalaga Thalaivan", fg='Blue', command=movie3)
movie3_buttonWindow = my_canvas.create_window(525, 400, anchor="nw", window=movie3_button)


movie4_poster = ImageTk.PhotoImage(Image.open(r".\movies\love_today.png"))
my_canvas.create_image(700, 150, image=movie4_poster, anchor="nw")
movie4_button = Button(parent2, text="Love Today", fg='Blue', command=movie4)
movie4_buttonWindow = my_canvas.create_window(725, 400, anchor="nw", window=movie4_button)


movie5_poster = ImageTk.PhotoImage(Image.open(r".\movies\ponniyin_selvan_part1.png"))
my_canvas.create_image(900, 150, image=movie5_poster, anchor="nw")
movie5_button = Button(parent2, text="Ponniyin Selvan: Part I", fg='Blue', command=movie5)
movie5_buttonWindow = my_canvas.create_window(900, 400, anchor="nw", window=movie5_button)


movie6_poster = ImageTk.PhotoImage(Image.open(r".\movies\don.png"))
my_canvas.create_image(1100, 150, image=movie6_poster, anchor="nw")
movie6_button = Button(parent2, text="Don", fg='Blue', command=movie6)
movie6_buttonWindow = my_canvas.create_window(1100, 400, anchor="nw", window=movie6_button)


movie7_poster = ImageTk.PhotoImage(Image.open(r".\movies\kaathuvaakula_rendu_kaadhal.png"))
my_canvas.create_image(1300, 150, image=movie7_poster, anchor="nw")
movie7_button = Button(parent2, text="Kaathuvaakula Rendu Kaadhal", fg='Blue', command=movie7)
movie7_buttonWindow = my_canvas.create_window(1300, 400, anchor="nw", window=movie7_button)


movie8_poster = ImageTk.PhotoImage(Image.open(r".\movies\KGF.png"))
my_canvas.create_image(100, 475, image=movie8_poster, anchor="nw")
movie8_button = Button(parent2, text="KGF Chapter:2", fg='Blue', command=movie8)
movie8_buttonWindow = my_canvas.create_window(100, 725, anchor="nw", window=movie8_button)
```

```
movie9_poster = ImageTk.PhotoImage(Image.open(r".\movies\vikram.png"))
my_canvas.create_image(300, 475, image=movie9_poster, anchor="nw")
movie9_button = Button(parent2, text="Vikram", fg='Blue', command=movie9)
movie9_buttonWindow = my_canvas.create_window(300, 725, anchor="nw", window=movie9_button)


movie10_poster = ImageTk.PhotoImage(Image.open(r".\movies\beast.png"))
my_canvas.create_image(500, 475, image=movie10_poster, anchor="nw")
movie10_button = Button(parent2, text="Beast", fg='Blue', command=movie10)
movie10_buttonWindow = my_canvas.create_window(500, 725, anchor="nw", window=movie10_button)


movie11_poster = ImageTk.PhotoImage(Image.open(r".\movies\valimai.png"))
my_canvas.create_image(700, 475, image=movie11_poster, anchor="nw")
movie11_button = Button(parent2, text="Valimai", fg='Blue', command=movie11)
movie11_buttonWindow = my_canvas.create_window(700, 725, anchor="nw", window=movie11_button)


movie12_poster = ImageTk.PhotoImage(Image.open(r".\movies\annaatthe.png"))
my_canvas.create_image(900, 475, image=movie12_poster, anchor="nw")
movie12_button = Button(parent2, text="Annaatthe", fg='Blue', command=movie12)
movie8_buttonWindow = my_canvas.create_window(900, 725, anchor="nw", window=movie12_button)


movie13_poster = ImageTk.PhotoImage(Image.open(r".\movies\master.png"))
my_canvas.create_image(1100, 475, image=movie13_poster, anchor="nw")
movie13_button = Button(parent2, text="Master", fg='Blue', command=movie13)
movie13_buttonWindow = my_canvas.create_window(1100, 725, anchor="nw", window=movie13_button)


movie14_poster = ImageTk.PhotoImage(Image.open(r".\movies\jai_bhim.png"))
my_canvas.create_image(1300, 475, image=movie14_poster, anchor="nw")
movie14_button = Button(parent2, text="Jai Bhim", fg='Blue', command=movie14)
movie14_buttonWindow = my_canvas.create_window(1300, 725, anchor="nw", window=movie14_button)

parent2.mainloop()
```

```
def select_seating():

    selected_seats = []

    parent3 = Tk()
    parent3.title("MyTheatreApp")
    parent3.geometry("1920x1080")
    parent3.iconbitmap(r".\icon\minecraft-icon-24.ico")

    theater_seats = ImageTk.PhotoImage(Image.open(r".\theatre\theater_seatings.png"))

    my_canvas = Canvas(parent3, width=1920, height=1080)
    my_canvas.pack(fill="both", expand=True)
    my_canvas.create_image(0, 0, image=theater_seats, anchor='nw')
    my_canvas.create_text(750, 100, text="Select Your Seats", font=("Times New Roman", 30))


    def button_clicked(seat):
        selected_seats.append(seat)


    def submit_seats():
        backend.write_seatData(selected_seats)


        def payment():

            pay = Tk()
            pay.title("ThankYou")
            pay.geometry("300x100")
            pay.iconbitmap(r".\icon\minecraft-icon-24.ico")
            amt = 250 * len(selected_seats)


            def ok_command():
                pay.destroy()
                thank_you()


            thankYou_label = Label(pay, text=f"Your Amount is: {amt}").pack()
            ok_button = Button(pay, text="Proceed to Pay", command=ok_command).pack()

        payment()

            def thank_you():

                thanks = Tk()
                thanks.title("ThankYou")
                thanks.geometry("300x100")
                thanks.iconbitmap(r".\icon\minecraft-icon-24.ico")


                def ok_command():

                    thanks.destroy()
                    parent3.destroy()


                thankYou_label = Label(thanks, text="Your Tickets have been booked Successfully!").pack()
                reciept_label = Label(thanks, text="Your reciept has been sent to your mail").pack()
                ok_button = Button(thanks, text="OK", command=ok_command).pack()


            def buttons():

                A1_button = Button(parent3, text="A1", command=lambda: button_clicked("A1"))
                A1_buttonWindow = my_canvas.create_window(196, 675, anchor="nw", window=A1_button)

                A2_button = Button(parent3, text="A2", command=lambda: button_clicked("A2"))
                A2_buttonWindow = my_canvas.create_window(323, 675, anchor="nw", window=A2_button)

                A3_button = Button(parent3, text="A3", command=lambda: button_clicked("A3"))
                A3_buttonWindow = my_canvas.create_window(456, 675, anchor="nw", window=A3_button)

                A4_button = Button(parent3, text="A4", command=lambda: button_clicked("A4"))
                A4_buttonWindow = my_canvas.create_window(589, 675, anchor="nw", window=A4_button)

                A5_button = Button(parent3, text="A5", command=lambda: button_clicked("A5"))
                A5_buttonWindow = my_canvas.create_window(722, 675, anchor="nw", window=A5_button)

                A6_button = Button(parent3, text="A6", command=lambda: button_clicked("A6"))
                A6_buttonWindow = my_canvas.create_window(855, 675, anchor="nw", window=A6_button)

                A7_button = Button(parent3, text="A7", command=lambda: button_clicked("A7"))
                A7_buttonWindow = my_canvas.create_window(988, 675, anchor="nw", window=A7_button)

                A8_button = Button(parent3, text="A8", command=lambda: button_clicked("A8"))
                A8_buttonWindow = my_canvas.create_window(1120, 675, anchor="nw", window=A8_button)
```

```
B1_button = Button(parent3, text="B1", command=lambda: button_clicked("B1"))
B1_buttonWindow = my_canvas.create_window(190, 564, anchor="nw", window=B1_button)

B2_button = Button(parent3, text="B2", command=lambda: button_clicked("B2"))
B2_buttonWindow = my_canvas.create_window(323, 564, anchor="nw", window=B2_button)

B3_button = Button(parent3, text="B3", command=lambda: button_clicked("B3"))
B3_buttonWindow = my_canvas.create_window(456, 564, anchor="nw", window=B3_button)

B4_button = Button(parent3, text="B4", command=lambda: button_clicked("B4"))
B4_buttonWindow = my_canvas.create_window(589, 564, anchor="nw", window=B4_button)

B5_button = Button(parent3, text="B5", command=lambda: button_clicked("B5"))
B5_buttonWindow = my_canvas.create_window(722, 564, anchor="nw", window=B5_button)

B6_button = Button(parent3, text="B6", command=lambda: button_clicked("B6"))
B6_buttonWindow = my_canvas.create_window(855, 564, anchor="nw", window=B6_button)

B7_button = Button(parent3, text="B7", command=lambda: button_clicked("B7"))
B7_buttonWindow = my_canvas.create_window(988, 564, anchor="nw", window=B7_button)

B8_button = Button(parent3, text="B8", command=lambda: button_clicked("B8"))
B8_buttonWindow = my_canvas.create_window(1120, 564, anchor="nw", window=B8_button)

C1_button = Button(parent3, text="C1", command=lambda: button_clicked("C1"))
C1_buttonWindow = my_canvas.create_window(190, 451, anchor="nw", window=C1_button)

C2_button = Button(parent3, text="C2", command=lambda: button_clicked("C2"))
C2_buttonWindow = my_canvas.create_window(323, 451, anchor="nw", window=C2_button)

C3_button = Button(parent3, text="C3", command=lambda: button_clicked("C3"))
C3_buttonWindow = my_canvas.create_window(456, 451, anchor="nw", window=C3_button)

C4_button = Button(parent3, text="C4", command=lambda: button_clicked("C4"))
C4_buttonWindow = my_canvas.create_window(589, 451, anchor="nw", window=C4_button)

C5_button = Button(parent3, text="C5", command=lambda: button_clicked("C5"))
C5_buttonWindow = my_canvas.create_window(722, 451, anchor="nw", window=C5_button)

C6_button = Button(parent3, text="C6", command=lambda: button_clicked("C6"))
C6_buttonWindow = my_canvas.create_window(855, 451, anchor="nw", window=C6_button)

C7_button = Button(parent3, text="C7", command=lambda: button_clicked("C7"))
C7_buttonWindow = my_canvas.create_window(988, 451, anchor="nw", window=C7_button)
```

```
C8_button = Button(parent3, text="C8", command=lambda: button_clicked("C8"))
C8_buttonWindow = my_canvas.create_window(1120, 451, anchor="nw", window=C8_button)

D1_button = Button(parent3, text="D1", command=lambda: button_clicked("D1"))
D1_buttonWindow = my_canvas.create_window(190, 338, anchor="nw", window=D1_button)

D2_button = Button(parent3, text="D2", command=lambda: button_clicked("D2"))
D2_buttonWindow = my_canvas.create_window(323, 338, anchor="nw", window=D2_button)

D3_button = Button(parent3, text="D3", command=lambda: button_clicked("D3"))
D3_buttonWindow = my_canvas.create_window(456, 338, anchor="nw", window=D3_button)

D4_button = Button(parent3, text="D4", command=lambda: button_clicked("D4"))
D4_buttonWindow = my_canvas.create_window(589, 338, anchor="nw", window=D4_button)

D5_button = Button(parent3, text="D5", command=lambda: button_clicked("D5"))
D5_buttonWindow = my_canvas.create_window(722, 338, anchor="nw", window=D5_button)

D6_button = Button(parent3, text="D6", command=lambda: button_clicked("D6"))
D6_buttonWindow = my_canvas.create_window(855, 338, anchor="nw", window=D6_button)

D7_button = Button(parent3, text="D7", command=lambda: button_clicked("D7"))
D7_buttonWindow = my_canvas.create_window(988, 338, anchor="nw", window=D7_button)

D8_button = Button(parent3, text="D8", command=lambda: button_clicked("D8"))
D8_buttonWindow = my_canvas.create_window(1120, 338, anchor="nw", window=D8_button)

E1_button = Button(parent3, text="E1", command=lambda: button_clicked("E1"))
E1_buttonWindow = my_canvas.create_window(190, 225, anchor="nw", window=E1_button)

E2_button = Button(parent3, text="E2", command=lambda: button_clicked("E2"))
E2_buttonWindow = my_canvas.create_window(323, 225, anchor="nw", window=E2_button)

E3_button = Button(parent3, text="E3", command=lambda: button_clicked("E3"))
E3_buttonWindow = my_canvas.create_window(456, 225, anchor="nw", window=E3_button)

E4_button = Button(parent3, text="E4", command=lambda: button_clicked("E4"))
E4_buttonWindow = my_canvas.create_window(589, 225, anchor="nw", window=E4_button)

E5_button = Button(parent3, text="E5", command=lambda: button_clicked("E5"))
E5_buttonWindow = my_canvas.create_window(722, 225, anchor="nw", window=E5_button)

E6_button = Button(parent3, text="E6", command=lambda: button_clicked("E6"))
E6_buttonWindow = my_canvas.create_window(855, 225, anchor="nw", window=E6_button)

                E7_button = Button(parent3, text="E7", command=lambda: button_clicked("E7"))
                E7_buttonWindow = my_canvas.create_window(988, 225, anchor="nw", window=E7_button)

                E8_button = Button(parent3, text="E8", command=lambda: button_clicked("E8"))
                E8_buttonWindow = my_canvas.create_window(1120, 225, anchor="nw", window=E8_button)

                submit_button = Button(parent3, text="Submit", command=submit_seats)
                submit_buttonWindow = my_canvas.create_window(655, 725, anchor="nw", window=submit_button)

            buttons()


            parent3.mainloop()

        parent1.mainloop()

    book_ticket()

frontend()
```

BACK END

```
# importing necessary module(s)
import csv

movie_Data = open("data.csv", "w", newline="")
wtr = csv.writer(movie_Data)
wtr.writerow(["Name", "Age", "Number of seats",  "Premium"])
movie_Data.close()

user_Data = open("user_data.csv", "w", newline="")
wtr = csv.writer(user_Data)
wtr.writerow(["Gmail", "Password"])
user_Data.close()


def input_data(name, age, seat_nos, premium):

    movie_Data = open("data.csv", "a", newline ="")
    wtr = csv.writer(movie_Data)
    wtr.writerow([name, age, seat_nos, premium])
    movie_Data.close()


def input_userData(gmail, password):
    user_Data = open("user_data.csv", "a", newline="")
    wtr = csv.writer(user_Data)
    wtr.writerow([gmail, password])


def write_newData(movie, genre, movie_lang, cast):
    old_Data = open("data.csv", 'r')
    rdr = csv.reader(old_Data)

    data = []

    for column in rdr:
        column.extend(['Movie', 'Genre', 'Movie Language', 'Cast'])
        data.append(column)
        break

    for column in rdr:
        column.extend([movie, genre, movie_lang, cast])
        data.append(column)

    old_Data.close()

    new_Data = open("data.csv", 'w', newline="")
    wtr = csv.writer(new_Data)


        new_Data = open("data.csv", 'w', newline="")
        wtr = csv.writer(new_Data)
        wtr.writerows(data)
        new_Data.close()

    def write_seatData(seats):
        old_Data = open("data.csv", 'r')
        rdr = csv.reader(old_Data)

        data = []

        for column in rdr:
            column.extend(["Selected Seats"])
            data.append(column)
            break

        for column in rdr:
            column.extend([seats])
            data.append(column)

        old_Data.close()

        new_Data = open("data.csv", 'w', newline="")
        wtr = csv.writer(new_Data)
        wtr.writerows(data)
        new_Data.close()
```

USER MANAGEMENT

```
# This program is to manage the data from the server side,
# although it is in the same directory as the client program.
# Here only csv file handling will be used to manage the data from
# "data.csv", into which we are inserting the data using frontend and backend

import csv

print("1. Delete record")
print("2. Sort based on language and update data")
print("3. Sort based on cast and update data")
print("4. Sort based on genre and update data")
print("5. Update record")

ch = int(input("Enter your choice: "))

def del_rec():

    f = open("data.csv","r", newline="")
    n = input("Enter the name whose record has to be deleted: ")
    l = []
    csv_reader = csv.reader(f)
    next(csv_reader)

    for i in csv_reader:
        if i[0] != n:
            l.append(i)

    f.close()

    g = open("data.csv","w",newline="")
    csv_writer = csv.writer(g)
    csv_writer.writerow(["Name","Age","Number of seats","Premium",
    "Movie","Genre","Movie Language","Cast"])
    csv_writer.writerows(l)

    g.close()
```


```
# This program is to manage the data from the server side,
# although it is in the same directory as the client program.
# Here only csv file handling will be used to manage the data from
# "data.csv", into which we are inserting the data using frontend and backend

import csv

print("1. Delete record")
print("2. Sort based on language and update data")
print("3. Sort based on cast and update data")
print("4. Sort based on genre and update data")
print("5. Update record")

ch = int(input("Enter your choice: "))

def del_rec():

    f = open("data.csv","r", newline="")
    n = input("Enter the name whose record has to be deleted: ")
    l = []
    csv_reader = csv.reader(f)
    next(csv_reader)

    for i in csv_reader:
        if i[0] != n:
            l.append(i)

    f.close()

    g = open("data.csv","w",newline="")
    csv_writer = csv.writer(g)
    csv_writer.writerow(["Name","Age","Number of seats","Premium",
    "Movie","Genre","Movie Language","Cast"])
    csv_writer.writerows(l)

    g.close()
```

```
def sort_lang():

    f = open("data.csv","r", newline="")
    csv_reader = csv.reader(f)
    next(csv_reader)
    l = []
    lang = []
    y = True
    for i in csv_reader:
        if i[6] not in lang:
            lang.append(i[6])
        l.append(i)
    sl=[]
    while y == True:
        for j in lang:
            for i in l:
                if i[6] == j:
                    sl.append(i)
                    l.remove(i)
        if len(l)==0:
            y = False

    f.close()

    g = open("data.csv","w",newline="")
    csv_writer = csv.writer(g)
    csv_writer.writerow(["Name","Age","Number of seats","Premium",
                         "Movie","Genre","Movie Language","Cast"])
    csv_writer.writerows(sl)

    g.close()



def sort_genre():

    f = open("data.csv","r", newline="")
    csv_reader = csv.reader(f)
    next(csv_reader)
    g = []
    genre = []
    y = True
    for i in csv_reader:
        if i[5] not in genre:
            genre.append(i[5])
        g.append(i)
    sg=[]
    while y == True:
        for j in genre:
            for i in g:
                if i[5] == j:
                    sg.append(i)
                    g.remove(i)
        if len(g) == 0:
            y = False

    f.close()

    h = open("data.csv","w",newline="")
    csv_writer = csv.writer(h)
    csv_writer.writerow(["Name","Age","Number of seats","Premium",
                         "Movie","Genre","Movie Language","Cast"])
    csv_writer.writerows(sg)

    h.close()
```

```
def sort_cast():

    f = open("data.csv","r", newline="")
    csv_reader = csv.reader(f)
    next(csv_reader)
    c = []
    cast = []
    y = True
    for i in csv_reader:
        if i[7][0] not in cast:
            cast.append(i[7][0])
        c.append(i)
    sc=[]
    while y == True:
        for j in cast:
            for i in c:
                if i[7][0] == j:
                    sc.append(i)
                    c.remove(i)
        if len(c)==0:
            y = False

    f.close()

    h = open("data.csv","w",newline="")
    csv_writer = csv.writer(h)
    csv_writer.writerow(["Name","Age","Number of seats","Premium",
                         "Movie","Genre","Movie Language","Cast"])
    csv_writer.writerows(sc)

    h.close()

if ch == 1:
    del_rec()
if ch == 2:
    sort_lang()
if  ch == 3:
    sort_cast()
if ch == 4:
    sort_genre()
if ch == 5:
    update_rec()
```

Scope For Improvement

- *Could have added more features to the application if more time was spent on it.*
- *Graphical quality of the application could have been improved with more understanding and knowledge about TKinter.*
- *Could have added a "Food Menu", which allows the user to order/purchase the snack they want to have during the movie.*
- *Could have added a proper bill which would have been sent to the user as a txt file, but due to time constraints, it was not possible.*
- *Could have added an QR Code to scan and pay in python, but due to time constraints, it was not possible.*

BIBLIOGRAPHY

- https://www.youtube.com/watch?v=YXPyB4XeYLA
- https://www.youtube.com/watch?v=WurCpmHtQc4
- https://www.reddit.com/r/Python/
- https://www.reddit.com/r/Tkinter/
- https://stackoverflow.com/
- https://www.geeksforgeeks.org/
- https://www.smartdraw.com/
- https://docs.python.org/3/library/tkinter.html