UNIVERSITI TEKNOLOGI MARA (UiTM),

MERBOK, KEDAH

FACULTY OF INFORMATION SCIENCE STUDIES,

COLLEGE OF COMPUTING,

INFROMATICS AND MEDIA STUDIES

DIPLOMA IN LIBRARY INFORMATICS (CDIM144

PROGRAMMING FOR LIBRARIES (IML208)

GROUP PROJECT- ONLINE CINEMA TICKET

CLASS: KIM1443B

PREPARED BY:

NAME	STUDENT ID
NUR IZZATI BINTI MOHD ABDUL MALEK	2022609496
NUR ANISA NAZIHAH BINTI KELANA	2022850266
ZARITH SAFIYAH INSYIRAH BINTI FAZLI	2022467502
NURAKMAL SYAHIRAH BINTI MUHAMAD SANI	2022820432

PREPARED FOR:

SIR AIRUL SHAZWAN BIN NORSHAHIMI

SUBMISSION DATE:

WEEK 14

GROUP PROJECT- ONLINE CINEMA TICKET

NUR IZZATI BINTI MOHD ABDUL MALEK

(2022609496)

NUR ANISA NAZIHAH BINTI KELANA

(2022850266)

ZARITH SAFIYAH INSYIRAH BINTI FAZLI

(2022467502)

NURAKMAL SYAHIRAH BINTI MUHAMAD SANI

(2022820432)

DIPLOMA IN LIBRARY INFORMATICS

FACULTY OF INFORMATION SCIENCE STUDIES,

COLLEGE OF COMPUTING,

INFROMATICS AND MEDIA STUDIES

17TH JANUARY 2024

Table of Content

1.0 Introduction

Regarding our assignment, we need four functions to be included in our program. The four functions are create, read, update, and delete. In this task, we need to make sure that these four can work well. In this assignment, we need to create a record to be entered into our database. In addition, we also need to make sure that we can use the delete button to delete the data and that we can use the update button to update the data that we have entered. In addition, we also need to make sure that the coding and our database can be well connected to each other. We also need to make sure that we can calculate the data entered by us.

For our project, we have chosen the title Cinema Tickets. The title we have chosen requires user information for the purchase of cinema tickets. In the user information contains name, age, gender, no phone, and email. Apart from that, our project also needs information about the movie showing that the user wants. In the showtime, the user needs to enter the date, movie name, time, and experience. Finally, the user also needs to make a choice about which seat they want to sit in the cinema. They can also choose how many children and adults there are. In our project, this also requires the calculation of all the total prices.

2.0 Problem Statement

Usually, most people like to go to the cinema with their families. So, they will go to the counter together to buy tickets. However, it is quite difficult for parents with small children to buy tickets at the counter. This will be difficult for them because they will need to queue to make a ticket purchase. If the person wants to make a choice, the other person has to queue for a long time just because they have to wait for the person to make a choice. Not only that, when they want to go to the cinema to see a movie but suddenly the tickets are sold out, it is just wasting their energy and time.

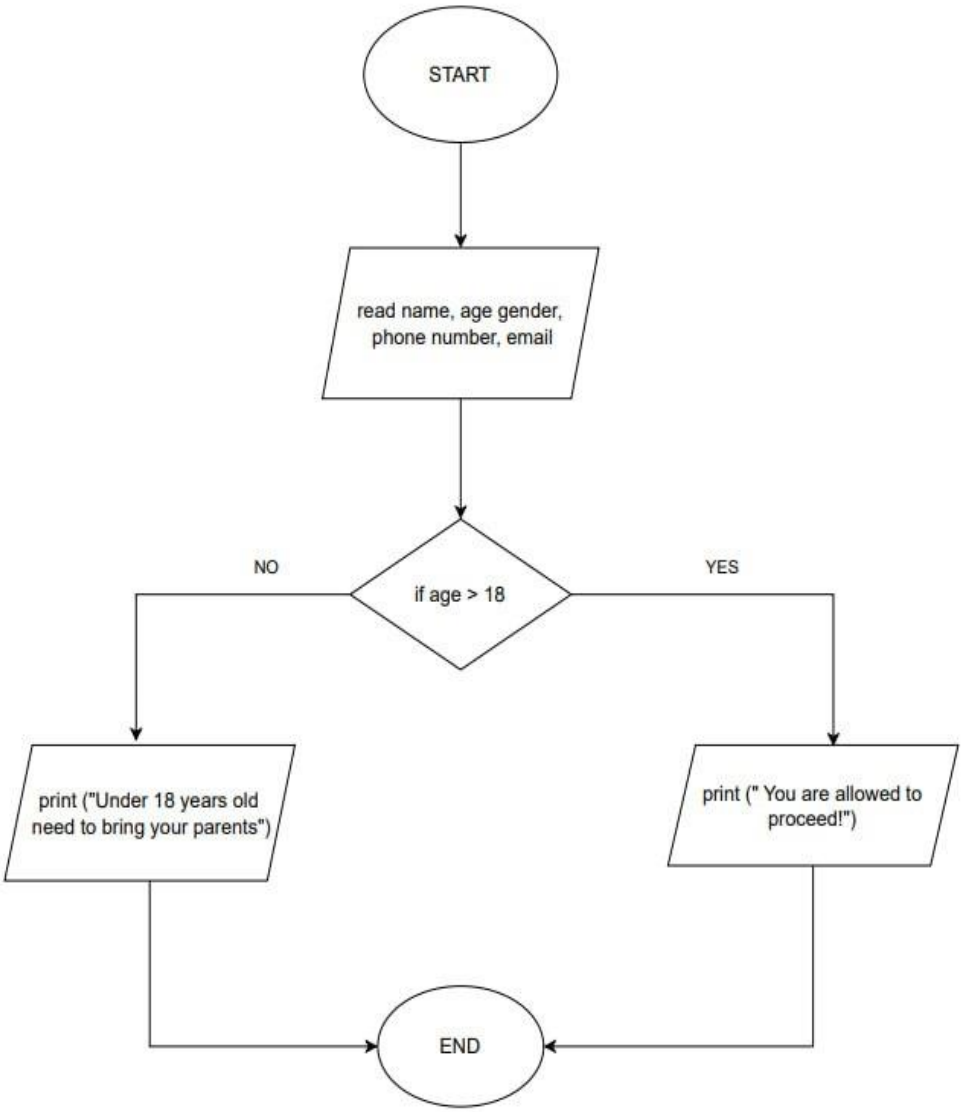
Therefore, in our assignment, we have found a way to overcome this problem. To make it easier for the public, they can buy cinema tickets online. Buyers need to enter their name and all their information online. For those who accidentally entered incorrect information, they can simply press the delete button to remove all their incorrect information. They can also choose the story they want without having to waste the time of people behind them waiting for them to make a decision to choose a movie. In addition, they can also choose which seat they want. Those who want to sit together can choose a pair of seats, and those with families can choose multiple seats. They can see and make a seat choice just by being at home. They can choose how many children and adults they want. Finally, they can make payments online without having to bother queuing or withdrawing money.

3.0 Objectives

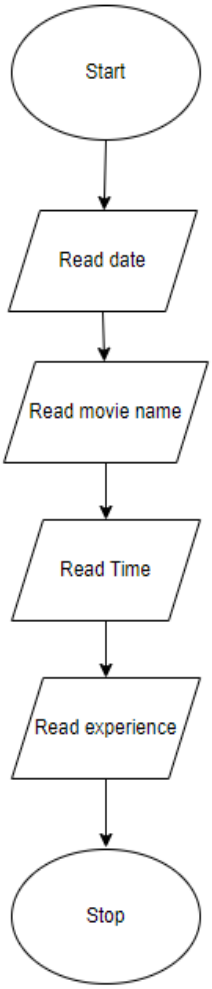
i. Make it easier for buyers to make choices.
ii. Buyers do not have to wait in long lines
iii. Make it easy for buyers to enter their information
iv. Save the buyer's time to make payment

4.0 Flowchart

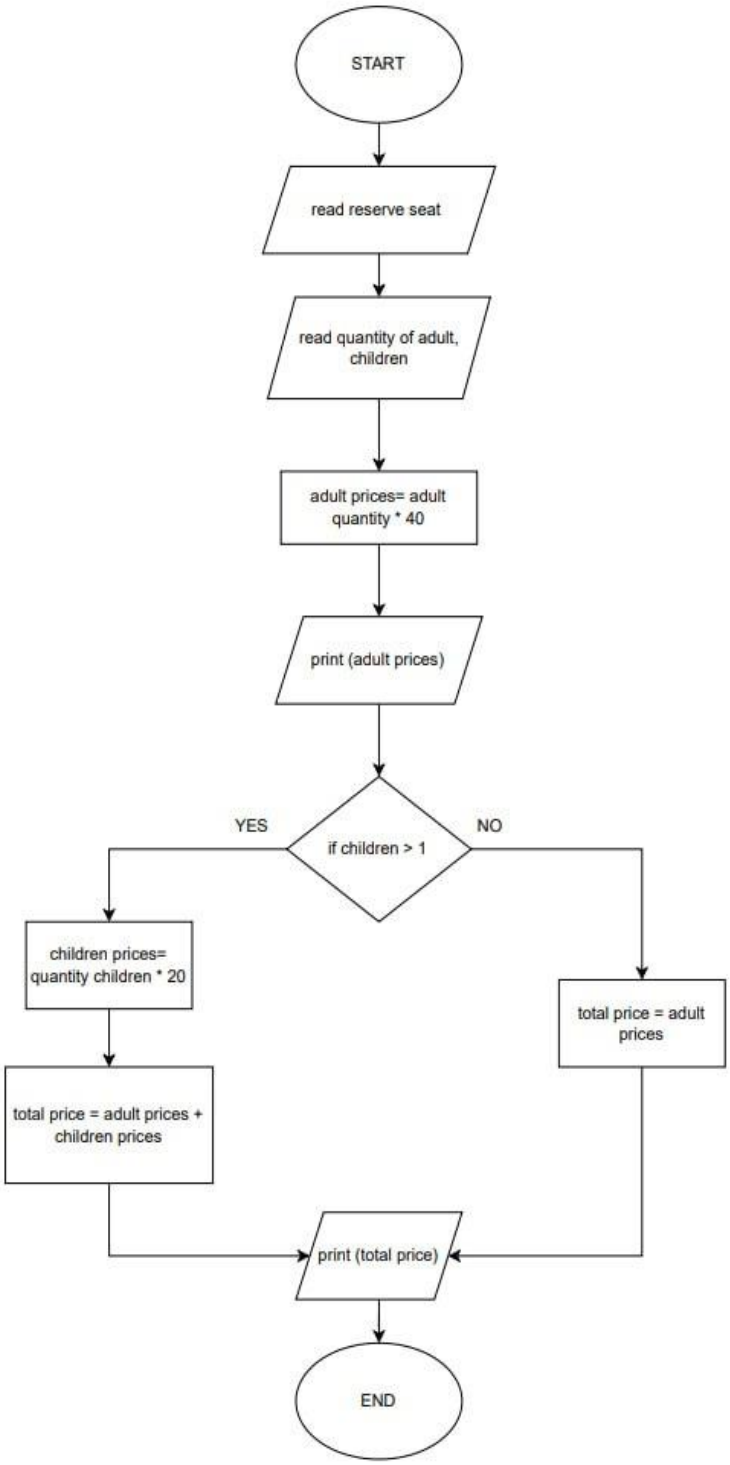
4.1 User Information



4.2 Showtime Selection



4.3 Seat on Cinema



5.0 Snapshot Code

5.1 Main Page

main_GSCticket.py ● registration_module.py showtime_movie.py reserve_seat.py icing_noyen.jpg

main_GSCticket.py > RootApp

```python
import tkinter as tk
from tkinter import ttk
import subprocess


class RootApp:

    def __init__(self, root):
        self.root = root
        self.root.title("GSC Ticket Booking")
        self.root.geometry("300x200")
        self.root.configure(background="#88AB8E")



        welcome_label = tk.Label(root, text='WELCOME TO THE GSC', font=("Times New Romans", 9, "bold"), bg="#88AB8E")
        welcome_label.pack()

        registration_button = tk.Button(root, text="User Registration", command=self.on_register_click, bg="#EEE7DA")
        registration_button.pack(pady=10)

```

main_GSCticket.py > ...

```python
        showtime_button = tk.Button(root, text="Showtime", command=self.on_showtime_click, bg="#EEE7DA")
        showtime_button.pack(pady=10)

        seat_button = tk.Button(root, text="Select Seat", command=self.on_select_seat_click, bg="#EEE7DA")
        seat_button.pack(pady=10)




    def on_register_click(self):
        subprocess.run(["python", "registration_module.py"])

    def on_showtime_click(self):
        subprocess.run(["python", "showtime_movie.py"])

    def on_select_seat_click(self):
        subprocess.run(["python", "reserve_seat.py"])
```

```python




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

5.2 User Registration

registration_module.py > register > create_widgets

```python
import tkinter
from tkinter import ttk
from tkinter import messagebox
from tkinter import *
import mysql.connector



# Connect to your MySQL database
mydb = mysql.connector.connect(
    host="localhost",
    user="root",
    password="",
    database="gsc_ticket"
)

# Create a cursor object to execute SQL queries
mycursor = mydb.cursor()


root = tkinter.Tk()

root.title("User Registration")
root.geometry("400x230")
root.configure(background="#88AB8E")


class register(tkinter.Frame):

```

```python

    def __init__(self, root):
        super().__init__(root)
        self.root = root
        self.pack()
        self.create_widgets()

    def create_widgets(self):
        # Add registration form widgets
        frame = tkinter.Frame(root)
        frame.pack()

        # user detail
        user_info_frame = tkinter.LabelFrame(frame, text="User Registration", bg="#88AB8E")
        user_info_frame.grid(row=0, column=0)

        self.name_label = tkinter.Label(user_info_frame, text=" Name", bg="#88AB8E")
        self.name_label.grid(row=0, column=0)
        self.name_entry = tkinter.Entry(user_info_frame, bg="#EEE7DA")
        self.name_entry.grid(row=1, column=0)

        self.age_label = tkinter.Label(user_info_frame, text=" Age", bg="#88AB8E")
        self.age_label.grid(row=0, column=1)
        self.age_spinbox = tkinter.Spinbox(user_info_frame, from_=0, to=55, bg="#EEE7DA")
        self.age_spinbox.grid(row=1, column=1)

        self.gender_label = tkinter.Label(user_info_frame, text="Gender", bg="#88AB8E")
        self.gender_label.grid(row=0, column=2)

```

```python

        # Use IntVar to store the state of the Checkbutton
        self.gender_var = tkinter.IntVar()
        self.gender_check_male = tkinter.Checkbutton(user_info_frame, text="Male", variable=self.gender_var, bg="#88AB8E")
        self.gender_check_male.grid(row=1, column=2)

        self.gender_check_female = tkinter.Checkbutton(user_info_frame, text="Female", bg="#88AB8E")
        self.gender_check_female.grid(row=1, column=4)

        self.no_phone_label = tkinter.Label(user_info_frame, text="Phone Number", bg="#88AB8E")
        self.no_phone_label.grid(row=2, column=0)
        self.no_phone_entry = tkinter.Entry(user_info_frame, bg="#EEE7DA")
        self.no_phone_entry.grid(row=3, column=0)

        self.email_label = tkinter.Label(user_info_frame, text="Email", bg="#88AB8E")
        self.email_label.grid(row=2, column=1)
        self.email_entry = tkinter.Entry(user_info_frame, bg="#EEE7DA")
        self.email_entry.grid(row=3, column=1)

        self.submit_button = tkinter.Button(root, text="Submit", background="gray", command=self.submit_data, bg="#EEE7DA")
        self.submit_button.pack(side=LEFT, padx=(10,5), pady=10)

        self.delete_button = tkinter.Button(root, text="Delete", command=self.delete_data, bg="#EEE7DA")
        self.delete_button.pack(side=LEFT, padx=5, pady=10)

        self.back_button = tkinter.Button(root, text="Back to Main Menu", command=self.back_to_main_menu, bg="#EEE7DA")
        self.back_button.pack(side=LEFT, padx=(5,10), pady=10)

    def delete_data(self):
```

```python
        mydb = mysql.connector.connect(
            host="localhost",
            user="root",
            password="",
            database="gsc_ticket"
        )

        mycursor = mydb.cursor()

        # Get the total count of rows in the database
        mycursor.execute("SELECT COUNT(*) FROM user_registration")
        count = mycursor.fetchone()[0]

        if count > 0:
            # Delete the last row in the database
            sql = "DELETE FROM user_registration ORDER BY name DESC LIMIT 1"
            mycursor.execute(sql)
            mydb.commit()
            print("Last record deleted.")
        else:
            print("No records to delete.")

        mycursor.close()
        mydb.close()
```

```python
    def submit_data(self):


        if not self.name_entry.get():
            tkinter.messagebox.showwarning(title="Error", message="Name is required.")
            return

        age_int = self.age_spinbox.get()

        age = int(age_int)

        if age >= 18:
            note_label = tkinter.Label (root, text="You are allowed to proceed!", font=("Times New Roman", 10, "italic"))
            note_label.pack(pady=5)
        else:
            note_label = tkinter.Label(root, text="Under 18 years old need to bring your parents.", font=("Times New Roman", 10, "italic"))
            note_label.pack(pady=5)

        print("Name:", self.name_entry.get())
        print("Age:",self.age_spinbox.get())
        print("Gender:",self.gender_var.get())
        print("Phone Number:",self.no_phone_entry.get())
        print("Email:",self.email_entry.get())


        # Get the value of the Checkbutton using get()
        gender = "Male" if self.gender_var.get() else "Female"

```

```python

        # To insert your Data to your database
        sql = "INSERT INTO user_registration (Name, Age, Gender, Phone_Number, Email) VALUES (%s, %s, %s, %s, %s)"
        val = (self.name_entry.get(), self.age_spinbox.get(), gender, self.no_phone_entry.get(), self.email_entry.get())
        mycursor.execute(sql, val)
        mydb.commit()


        mycursor.close()
        mydb.close()


    def back_to_main_menu(self):
        self.root.destroy()


if __name__== "__main__":
    app = register(root)
    root.mainloop()
```

5.3 Showtime Selection

showtime_movie.py > Showtime_movie > create_widgets

```python
import tkinter
from tkinter import ttk
from tkcalendar import DateEntry
from tkinter import *
import mysql.connector

# Connect to your MySQL database
mydb = mysql.connector.connect(
    host="localhost",
    user="root",
    password="",
    database="gsc_ticket"
)

# Create a cursor object to execute SQL queries
mycursor = mydb.cursor()

root = tkinter.Tk()
root.title("Showtime Movie")
root.configure(background="#88AB8E")

class Showtime_movie(tkinter.Frame):

    def __init__(self, root):
        super().__init__(root)
        self.root = root
        self.pack()
        self.create_widgets()

```

```python
    def create_widgets(self):
        # Add registration form widgets
        frame = tkinter.Frame(root, background="#88AB8E")
        frame.pack()

        showtime_frame = tkinter.LabelFrame(frame, text = "Showtime", bg="#88AB8E")
        showtime_frame.grid (row= 0, column= 0, padx=5, pady= 5)

        def pick_date():
            top = tkinter.Toplevel(showtime_frame)
            cal = DateEntry(top, font="Arial 8", selectmode="day", locale="en_US")
            cal.pack(fill="both", expand=True)

            def on_date_selected():
                self.date_label.set(cal.get_date())
                top.destroy()

            button_ok = tkinter.Button(top, text="OK", command=on_date_selected)
            button_ok.pack()

        self.date_label = tkinter.StringVar()
        self.date_label.set("Select Date")
        self.date_entry = tkinter.Entry(showtime_frame,textvariable=self.date_label)
        self.date_entry.grid(row=2, column=0, padx=10, pady=10)

        self.button_pick_date = tkinter.Button(showtime_frame, text="Pick Date", background= "#EEE7DA", command=pick_date)
        self.button_pick_date.grid(row=3, column=0, padx=5, pady=5)

```

```python
        #button to pick date
        self.movie_label=tkinter.Label(showtime_frame,text="Movie Name", bg="#88AB8E")
        self.movie_label.grid(row=1, column= 1, padx=5, pady=5)
        self.mv_combobox = ttk.Combobox (showtime_frame, values = ["Aquaman", "Endless Journey"])
        self.mv_combobox.grid(row=2, column= 1, padx=5, pady=5)

       # list time
        self.time_label=tkinter.Label(showtime_frame, text='Time', bg="#88AB8E")
        self.time_label.grid(row=1, column=2, padx=5, pady=5)

        self.options_list = ["10:00 a.m - 12:15 p.m", "4:00 p.m - 6:00 p.m", "8:00 p.m-10:00 p.m", "9:00 p.m -12:00 p.m"]

        self.value_inside = tkinter.StringVar(showtime_frame)
        self.value_inside.set("Select Your Time")

        self.time_menu = tkinter.OptionMenu(showtime_frame, self.value_inside,*self.options_list)
        self.time_menu.grid(row=2,column=2, padx=5, pady=5)
        self.time_menu.config(bg="#EEE7DA")
        self.time_menu["menu"].config(bg="#EEE7DA")

        self.exp_label=tkinter.Label(showtime_frame, text='Experience', bg="#88AB8E")
        self.exp_label.grid(row=1, column= 3, padx=5, pady=5)
        self.exp_combobox = ttk.Combobox (showtime_frame, values = ["Standard Class","SkyBox","PREMIERE","Onyx"])
        self.exp_combobox.grid(row=2, column= 3, padx=5, pady=5)

        # Prices List by using textbox
        self.note_text = tkinter.Text(showtime_frame, height=15, width=30,bg="#EEE7DA")
        self.note_text.grid(row= 0, column=1, padx=5,pady=5, sticky="nswe")

```

```python
        # The defined list by using pricebox
        self.note_text.insert(tkinter.END, "Movie List and Time\n\n")
        self.note_text.insert(tkinter.END, " Aquaman\n Time : 10:00 a.m - 12:15 p.m\n\n")
        self.note_text.insert(tkinter.END, " Endless Journey\n Time : 4:00 p.m - 6:00 p.m\n\n")
        self.note_text.insert(tkinter.END, " Aquaman\n Time : 8:00 p.m-10:00 p.m \n\n")
        self.note_text.insert(tkinter.END, " Endless Journey\n Time : 9:00 p.m -12:00 p.m\n\n")
        self.note_text.configure(state='disabled')

        def submit_data():
            sql = "INSERT INTO showtime_selection (Date_label,Movie_label,Time_label,Experience_label) VALUES (%s, %s, %s, %s)"
            val = (self.date_entry.get(),self.mv_combobox.get(),self.value_inside.get(),self.exp_combobox.get())
            mycursor.execute(sql, val)
            mydb.commit()

        def update():
            date = self.date_entry.get()
            movie = self.mv_combobox.get()
            time = self.value_inside.get()
            experience = self.exp_combobox.get()

            mydb = mysql.connector.connect(
            host="localhost",
            user="root",
            password="",
            database="gsc_ticket"
            )
            mycursor = mydb.cursor()

```

```python
            # Get the total count of rows in the database
            mycursor.execute("SELECT COUNT(*) FROM showtime_selection")
            count = mycursor.fetchone()[0]

            if count > 0:
                # Update the last row in the database
                sql = "UPDATE showtime_selection SET Date_label= %s, Movie_label = %s, Time_label= %s,Experience_label= %s WHERE Date_label
                val = (date,movie,time,experience)
                mycursor.execute(sql, val)
                mydb.commit()
                print("Update Record.")
            else:
                print("No records to update.")


        def back_to_main_menu():
            self.root.destroy()


        self.submit_button = tkinter.Button(root, text='Submit', command=submit_data, bg="#EEE7DA")
        self.submit_button.pack(side=LEFT,padx=5, pady=10)

        self.update_button = tkinter.Button(root, text='Update', command=update, bg="#EEE7DA")
        self.update_button.pack(side=LEFT,padx=5, pady=10)

        self.back_button = tkinter.Button(root, text='Back to Main Menu', command=back_to_main_menu, bg="#EEE7DA")
        self.back_button.pack(side=LEFT,padx=5, pady=10)
```

```python
Date_label= %s, Movie_label = %s, Time_label= %s,Experience_label= %s WHERE Date_label= %s"
```

```python


if __name__== "__main__":
    app = Showtime_movie(root)
    root.mainloop()


```

5.4 Seat on Cinema

reserve_seat.py > Reserve_seat > create_widgets

```python
# Import the required libraries
import tkinter
from tkinter import ttk
from tkinter import *
from PIL import Image, ImageTk
import mysql.connector

mydb = mysql.connector.connect(
    host="localhost",
    user="root",
    password="",
    database="gsc_ticket"
    )


mycursor = mydb.cursor()

root = tkinter.Tk()
root.title("Selected Seat")
root.configure(background="#88AB8E")
```

```python


class Reserve_seat(tkinter.Frame):

    def __init__(self, root):
        super().__init__(root)
        self.root = root
        self.my_img = None
        self.pack()
        self.create_widgets()

    def calculate_total(self):
        quantity_adult = int(self.adult_quantity_box.get())
        adult_prices= quantity_adult * 40

        children= int(self.children_quantity_box.get())

        if children > 1:
            child_prices= children * 20
            total_price= (adult_prices + child_prices)

        else:
            total_price= (adult_prices)

        self.total_output_label.config(text=f"Total Price: RM {total_price}")

```

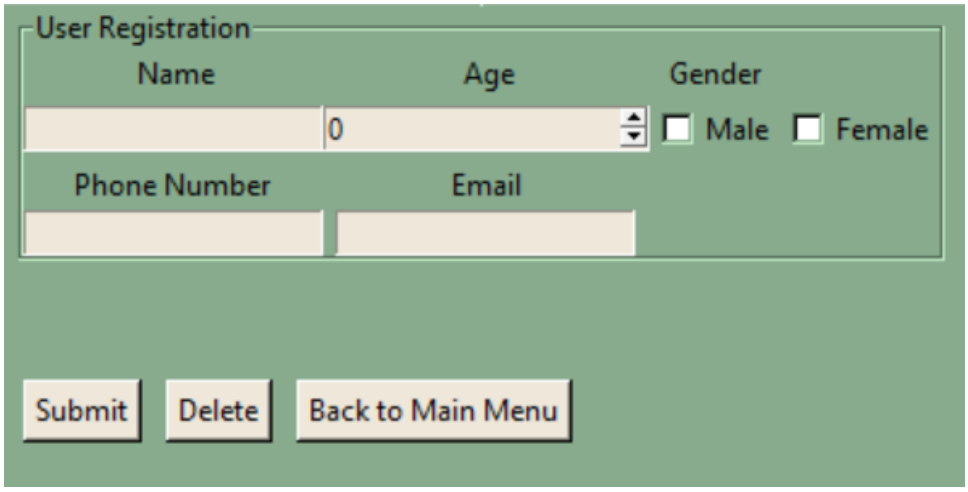
```python
    def save_data(self):
        reserve_seat = self.type_of_seat_combo.get()
        quantity_adult = int(self.adult_quantity_box.get())
        quantity_children = int(self.children_quantity_box.get())

        self.calculate_total()

        # To insert data into database, modify the following lines:
        sql = "INSERT INTO customer_seat_cinema (reserve_seat, quantity_of_adult , quantity_of_children, total_price) VALUES (%s, %s, %s, %s)"
        val = (self.type_of_seat_combo.get(), self.adult_quantity_box.get(), self.children_quantity_box.get(), self.total_output_label.cget("text"))

        mycursor.execute(sql, val)
        mydb.commit()


    def create_widgets(self):

        image=Image.open('icing_noyen.jpg')
        img=image.resize((350, 250))
        self.my_img=ImageTk.PhotoImage(img)
        label=Label(self.root, image=self.my_img)
        label.pack()

        self.frame= tkinter.Frame(self.root, bg="#88AB8E")
        self.frame.pack()

        self.reserve_seat_frame =tkinter.LabelFrame(self.frame, text="Seat on Cinema", bg="#88AB8E")
```

```python
        self.reserve_seat_frame.grid(row= 1, column=0, padx=20, pady=20)


        self.reserve_seat_label= tkinter.Label(self.reserve_seat_frame, text="Reserve Seat", bg="#88AB8E")
        self.reserve_seat_label.grid(row=0, column= 1,padx=5, pady=5)
        number_of_seat=tkinter.StringVar()
        self.type_of_seat_combo = ttk.Combobox(self.reserve_seat_frame, values=["A1", "A2", "A3", "A4", "A5","A6","B1","B2","B3","B4",
        "B5","B6","B7","B8"])
        self.type_of_seat_combo.grid(row=1, column=1, padx=5, pady=5)

        self.quantity_adult_label = tkinter.Label(self.reserve_seat_frame, text ='Adult', bg="#88AB8E")
        self.quantity_adult_label.grid(row=0, column=3, padx=5, pady=5)
        self.adult_quantity_box = tkinter.Spinbox(self.reserve_seat_frame, from_= 1, to = 30, bg="#EEE7DA")
        self.adult_quantity_box.grid(row=1, column=3, padx=5, pady=5)

        self.quantity_children_label = tkinter.Label(self.reserve_seat_frame, text ='Children', bg="#88AB8E")
        self.quantity_children_label.grid(row=0, column=2, padx=5, pady=5)
        self.children_quantity_box = tkinter.Spinbox(self.reserve_seat_frame, from_= 0, to = 10, bg="#EEE7DA")
        self.children_quantity_box.grid(row=1, column=2, padx=5, pady=5)

        self.calculate_button = tkinter.Button(text = "Calculate", width=15, command=self.calculate_total, bg="#EEE7DA")
        self.calculate_button.pack(side=LEFT, padx=(10,5), pady=10)

        self.save_button = tkinter.Button(text = "Submit", width=15, command=self.save_data, bg="#EEE7DA")
        self.save_button.pack(side=LEFT, padx=5, pady=10)

        self.total_output_label = tkinter.Label(self.reserve_seat_frame, text="", bg="#88AB8E")
        self.total_output_label.grid(row=3, column=2, padx=5, pady=5)

```

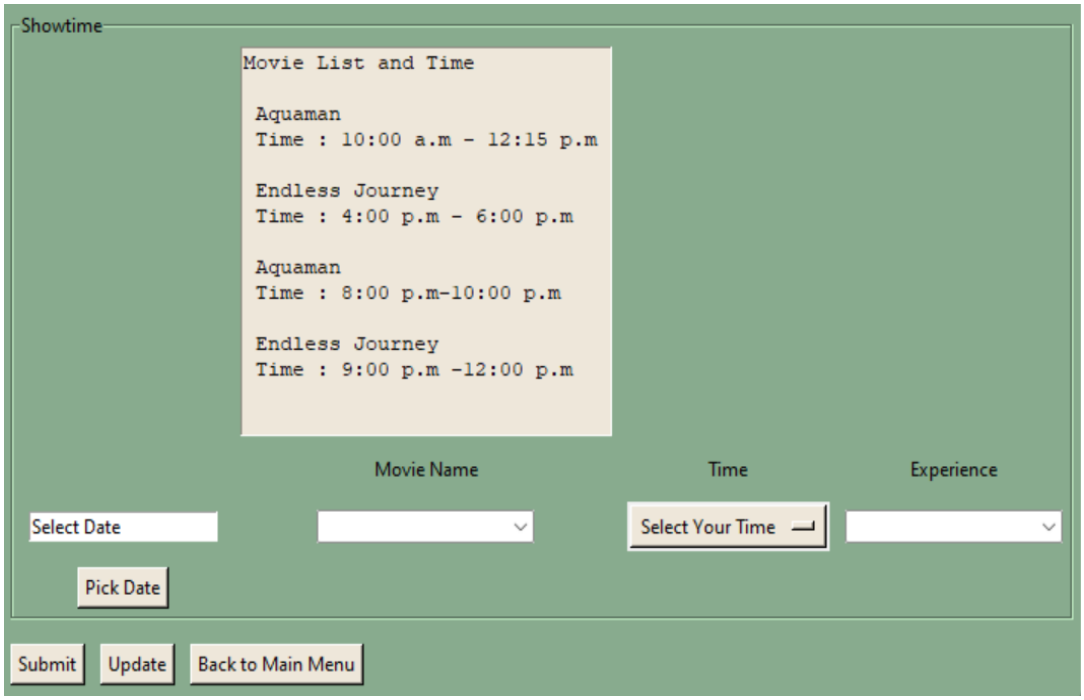
```python
        self.back_button = tkinter.Button(text="Back to Main Menu", width=15, command=self.back_to_main_menu, bg="#EEE7DA")
        self.back_button.pack(side=LEFT, padx=(5,10), pady=10)


    def back_to_main_menu(self):
            self.root.destroy()

if __name__== "__main__":
    app = Reserve_seat(root)
    root.mainloop()
```

6.0 Snapshot GUI

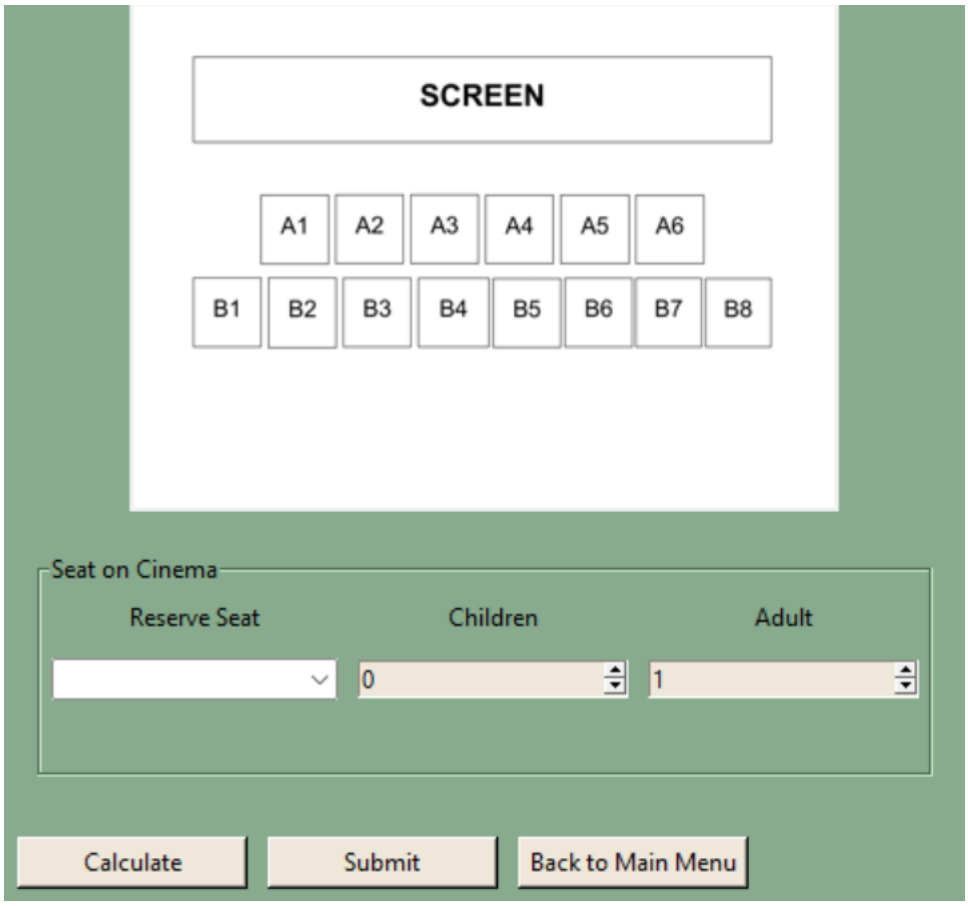
6.1 User Information



6.2 Showtime Selection

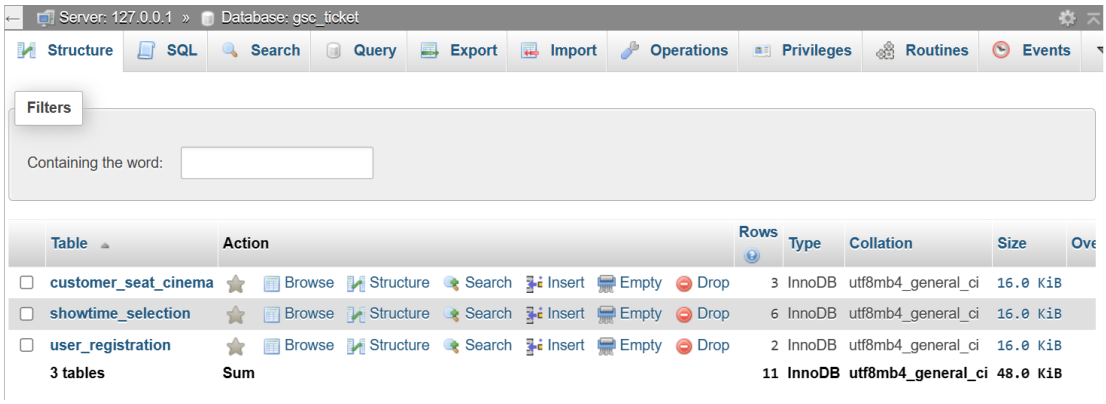


6.3 Seat on Cinema

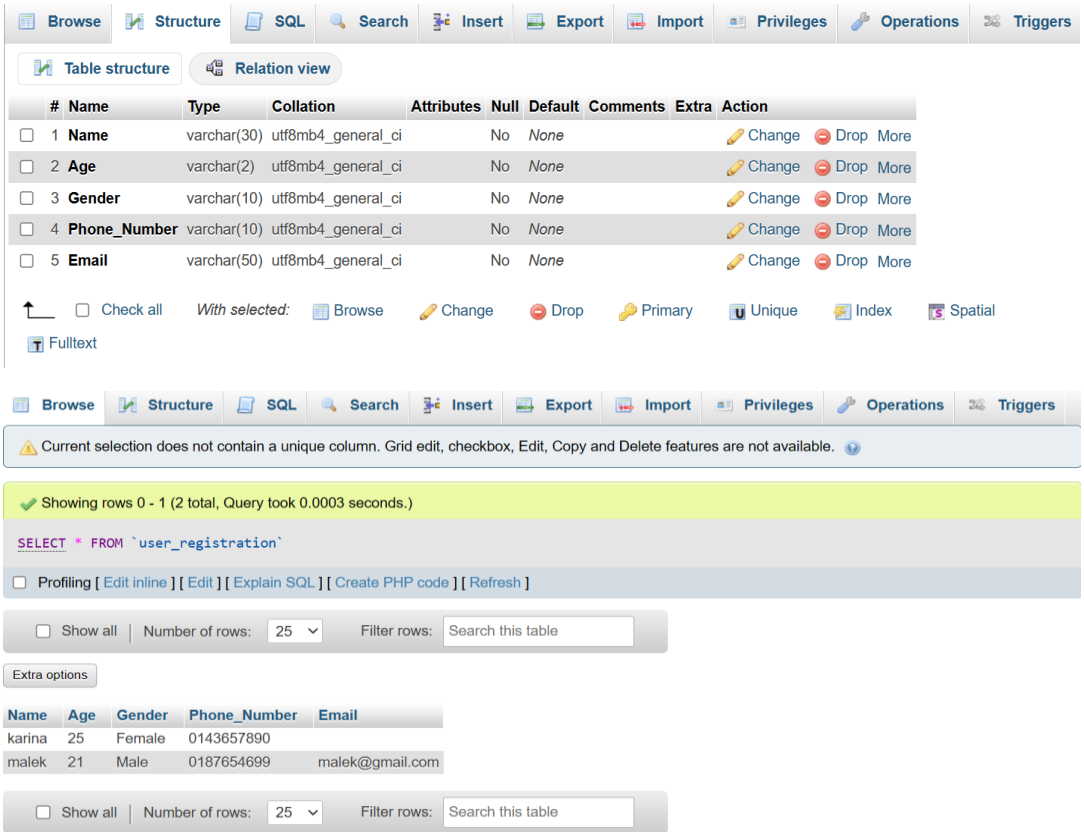


7.0 Snapshot Database

Database name and table name



7.1 User Registration Table



7.2 Showtime Table

Server: 127.0.0.1 » Database: gsc_ticket » Table: showtime_selection

Browse | Structure | SQL | Search | Insert | Export | Import | Privileges | Operations | Triggers

Table structure | Relation view

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	Date_label	varchar(30)	utf8mb4_general_ci		No	None			Change Drop More
2	Movie_label	varchar(30)	utf8mb4_general_ci		No	None			Change Drop More
3	Time_label	varchar(30)	utf8mb4_general_ci		No	None			Change Drop More
4	Experience_label	varchar(30)	utf8mb4_general_ci		No	None			Change Drop More

Check all *With selected:* Browse Change Drop Primary Unique Index Spatial Fulltext

Server: 127.0.0.1 » Database: gsc_ticket » Table: showtime_selection

Browse | Structure | SQL | Search | Insert | Export | Import | Privileges | Operations | Triggers

Current selection does not contain a unique column. Grid edit, checkbox, Edit, Copy and Delete features are not available.

Showing rows 0 - 5 (6 total, Query took 0.0004 seconds.)

```
SELECT * FROM `showtime_selection`
```

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

Show all | Number of rows: 25 Filter rows: Search this table

Extra options

Date_label	Movie_label	Time_label	Experience_label
2024-01-17	Endless Journey	10:00 a.m - 12:15 p.m	SkyBox
2024-01-17	Endless Journey	10:00 a.m - 12:15 p.m	Onyx
2024-01-17	Endless Journey	9:00 p.m -12:00 p.m	PREMIERE
2024-01-19	Endless Journey	9:00 p.m -12:00 p.m	PREMIERE
2024-01-22	Aquaman	10:00 a.m - 12:15 p.m	SkyBox
2024-01-22	Endless Journey	4:00 p.m - 6:00 p.m	Standard Class

7.3 Seat on Cinema Table

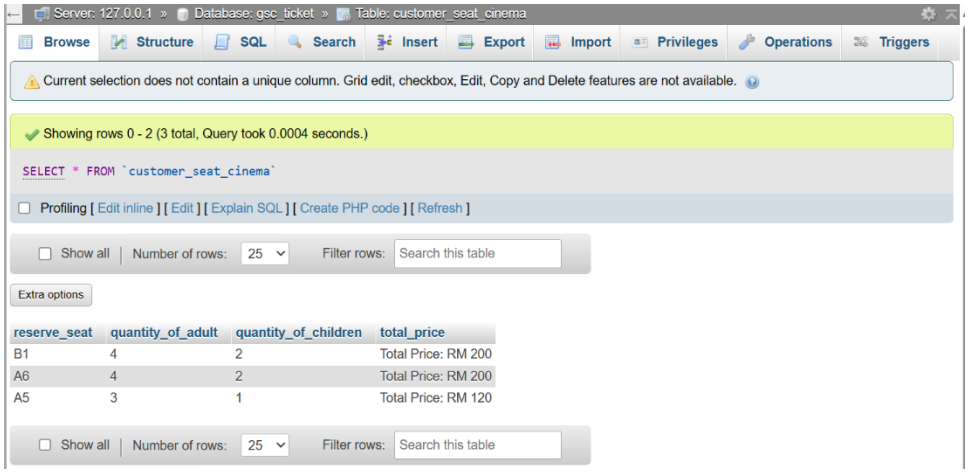
Server: 127.0.0.1 » Database: gsc_ticket » Table: customer_seat_cinema

Browse | Structure | SQL | Search | Insert | Export | Import | Privileges | Operations | Triggers

Table structure | Relation view

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	reserve_seat	varchar(20)	utf8mb4_general_ci		No	None			Change Drop More
2	quantity_of_adult	varchar(20)	utf8mb4_general_ci		No	None			Change Drop More
3	quantity_of_children	varchar(20)	utf8mb4_general_ci		No	None			Change Drop More
4	total_price	varchar(20)	utf8mb4_general_ci		No	None			Change Drop More

Check all *With selected:* Browse Change Drop Primary Unique Index Spatial Fulltext



8.0 Conclusion

In conclusion, our group succeeded in producing a GUI that meets CRUD criteria. Indirectly, by doing all these tasks, we can learn some things about it. The information entered in the database can give us knowledge about the output function of our GUI. The coding that we produce also requires a high level of knowledge. At first, when we did our assignment, there were various problems that we faced. Nevertheless, after we discussed and explored together, we were able to solve this problem. We hope that the papers we write today will be useful in the future.