PROJECT CODE :

```
import tkinter as tk


class Stadium:
    def _init_(self, capacity, vip_capacity, upper_capacity, lower_capacity):
        self.capacity = capacity
        self.available_seats = capacity
        self.vip_capacity = vip_capacity
        self.available_vip_seats = vip_capacity
        self.upper_capacity = upper_capacity
        self.available_upper_seats = upper_capacity
        self.lower_capacity = lower_capacity
        self.available_lower_seats = lower_capacity

        self.seats = {
            "VIP": [False] * vip_capacity,
            "Upper": [False] * upper_capacity,
            "Lower": [False] * lower_capacity
        }

    def sell_ticket(self, seat_class, name, age):
        if seat_class not in self.seats:
            print("Invalid seat class.")
            return

        seats = self.seats[seat_class]
        capacity = getattr(self, f"{seat_class.lower()}_capacity")
        available_seats = getattr(self, f"available_{seat_class.lower()}_seats")

        num_seats = int(input(f"How many {seat_class} seats would you like to book? "))
```

```
    if available_seats >= num_seats:
        booked_seats = []
        for i in range(capacity):
            if not seats[i]:  # Check if seat is available
                seats[i] = True  # Mark seat as occupied
                booked_seats.append(i + 1)
                setattr(self, f"available_{seat_class.lower()}_seats", available_seats - 1)
                if len(booked_seats) == num_seats:
                    break

        booking_info = f"Booking for {seat_class} seats {booked_seats} for {name} (Age: {age})\n"
        with open("booking_status.txt", "a") as file:
            file.write(booking_info)

        print(f"Tickets sold for {seat_class} seats {booked_seats} for {name} (Age: {age})")
        return True
    else:
        print(f"Sorry, not enough {seat_class.lower()} seats available.")
        return False

def return_ticket(self, seat_class, seat_number):
    if seat_class not in self.seats:
        print("Invalid seat class.")
        return False

    seats = self.seats[seat_class]
    capacity = getattr(self, f"{seat_class.lower()}_capacity")

    if 1 <= seat_number <= capacity:
        if seats[seat_number - 1]:  # Check if seat is occupied
            seats[seat_number - 1] = False  # Mark seat as available
```

```
            setattr(self, f"available_{seat_class.lower()}seats", getattr(self,
f"available{seat_class.lower()}_seats") + 1)


            with open("booking_status.txt", "r") as file:
                lines = file.readlines()
            with open("booking_status.txt", "w") as file:
                for line in lines:
                    if f"{seat_class} seat {seat_number}" not in line:
                        file.write(line)


            print(f"Ticket returned for {seat_class} seat {seat_number}")
            return True
        else:
            print(f"{seat_class} seat {seat_number} is already available.")
            return False
    else:
        print("Invalid seat number.")
        return False


  def check_availability(self, seat_class=None):
    if seat_class:
        if seat_class not in self.seats:
            print("Invalid seat class.")
            return
        return getattr(self, f"available_{seat_class.lower()}_seats")
    else:
        total_available_seats = sum(getattr(self, f"available_{seat_class.lower()}_seats") for seat_class
in self.seats)
        return total_available_seats


class StadiumGUI:
  def _init_(self, root):
```

```
        self.root = root
        self.stadium = Stadium(capacity=300, vip_capacity=50, upper_capacity=100,
lower_capacity=150)
        self.setup_gui()

    def setup_gui(self):
        self.root.title("Stadium Seating Management")

        # Sidebar
        self.sidebar = tk.Frame(self.root, bg="lightgray", width=200)
        self.sidebar.pack(side="left", fill="y")

        # Menu buttons
        menu_items = [
            ("Sell Ticket (VIP)", self.sell_vip_ticket),
            ("Sell Ticket (Upper Class)", self.sell_upper_ticket),
            ("Sell Ticket (Lower Class)", self.sell_lower_ticket),
            ("Return Ticket (VIP)", self.return_vip_ticket),
            ("Return Ticket (Upper Class)", self.return_upper_ticket),
            ("Return Ticket (Lower Class)", self.return_lower_ticket),
            ("Check Availability (VIP)", self.check_vip_availability),
            ("Check Availability (Upper Class)", self.check_upper_availability),
            ("Check Availability (Lower Class)", self.check_lower_availability),
            ("Check Total Availability", self.check_total_availability),
            ("Exit", self.root.quit)
        ]

        for text, command in menu_items:
            btn = tk.Button(self.sidebar, text=text, width=20, command=command)
            btn.pack(pady=5)
```

```
def sell_vip_ticket(self):
    name = input("Enter your name: ")
    age = input("Enter your age: ")
    if self.stadium.sell_ticket("VIP", name, age):
        print("Ticket sold successfully.")
    else:
        print("Ticket could not be sold.")

def sell_upper_ticket(self):
    name = input("Enter your name: ")
    age = input("Enter your age: ")
    if self.stadium.sell_ticket("Upper", name, age):
        print("Ticket sold successfully.")
    else:
        print("Ticket could not be sold.")

def sell_lower_ticket(self):
    name = input("Enter your name: ")
    age = input("Enter your age: ")
    if self.stadium.sell_ticket("Lower", name, age):
        print("Ticket sold successfully.")
    else:
        print("Ticket could not be sold.")

def return_vip_ticket(self):
    seat_number = int(input("Enter VIP seat number to return ticket: "))
    if self.stadium.return_ticket("VIP", seat_number):
        print("Ticket returned successfully.")
    else:
        print("Ticket could not be returned.")
```

```
    def return_upper_ticket(self):
        seat_number = int(input("Enter Upper Class seat number to return ticket: "))
        if self.stadium.return_ticket("Upper", seat_number):
            print("Ticket returned successfully.")
        else:
            print("Ticket could not be returned.")

    def return_lower_ticket(self):
        seat_number = int(input("Enter Lower Class seat number to return ticket: "))
        if self.stadium.return_ticket("Lower", seat_number):
            print("Ticket returned successfully.")
        else:
            print("Ticket could not be returned.")

    def check_vip_availability(self):
        print("Available VIP seats:", self.stadium.check_availability("VIP"))

    def check_upper_availability(self):
        print("Available Upper Class seats:", self.stadium.check_availability("Upper"))

    def check_lower_availability(self):
        print("Available Lower Class seats:", self.stadium.check_availability("Lower"))

    def check_total_availability(self):
        print("Total Available seats:", self.stadium.check_availability())

root = tk.Tk()
app = StadiumGUI(root)
root.mainloop()
```