

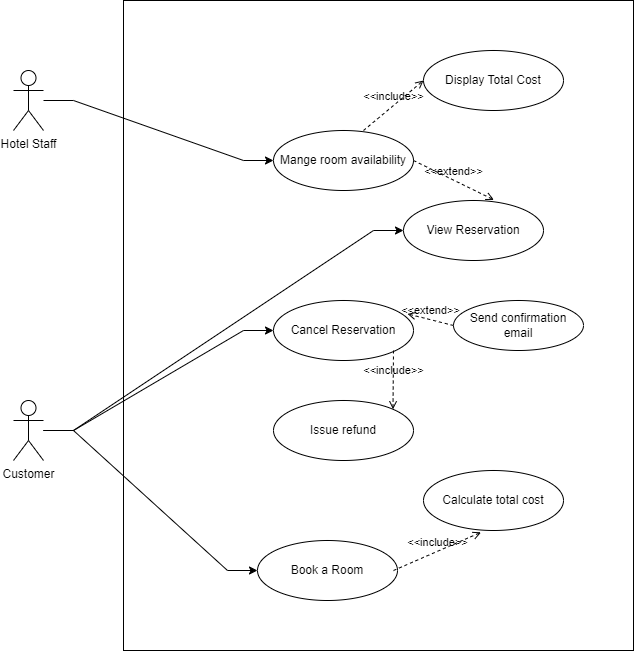
ICS220

Hotel Reservation System



September 25, 2024

**1. Use case Diagram**



**Use-Case Descriptions:**

**Book a Room**

**Primary Actor:** Customer

Goal: This customers reserves a room for dates and preferences of his or her choice.

**Includes:**

Calculate Total Cost: Calculates the amount spent on the food automatically depending on room charge and number of nights.

**Extends**: None.

**View Reservation**

**Primary Actor:** Customer

Goal: Customer gets to see the current details of their reserved area.

**Includes:**

Display Total Cost: Present the complete cost of the reservation.

**Extends:** None.

**Cancel Reservation**

**Primary Actor:** Customer

Goal: Customer may choose to cancel their previous appointment.

**Includes:**

Issue Refund: Another one is that it can automatically issue refund on cancelled reservation.

**Extends:**

Send Confirmation Email: After cancellation is complete the system sends the customer a confirmation email.

**Manage Room Availability**

**Primary Actor:** Hotel Staff

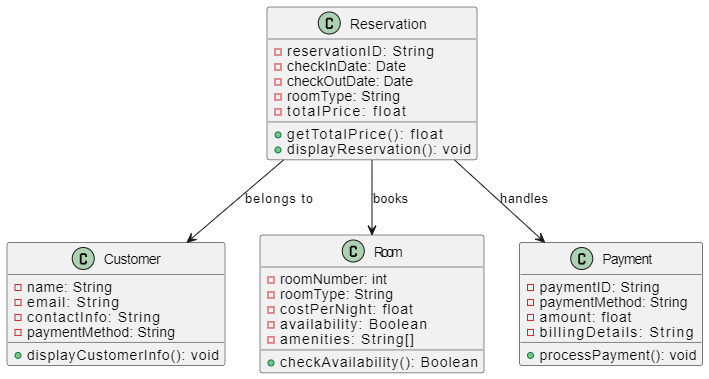
Goal: Schedule rooms according to the number of guests checked-in and checked-out.

**Includes:** None.

**Extends:**

View Reservation: Hotel customer reservations can be checked by the staff to ascertain the new status of the rooms.

**2. Class Diagram**



**Class Diagram Description**

**Reservation Class**

**Attributes:**

reservationID: Refers to a code given to every reservation made with the firm.

checkInDate: Month, date and year the customer will pay his check in.

checkOutDate: The date at which the customer will be checking out.

roomType: Whether the room was sold as a single unit or not (for example, “2 Queen Beds”).

totalPrice: Refers to the total cost of the specific room type for the number of days of the reservation.

**Methods:**

getTotalPrice(): Returns the total price of the reserved product on the basis of number of days taken for reservation.

displayReservation(): Shows necessary information about the reservation regarding the room type, dates, cost.

Associations: A reservation links a customer to a specific room and is paid for.

**Customer Class**

**Attributes:**

name: The name of the customer who made the reservation is used.

email: The email of the customer for purpose of communication.

contactInfo: The full name, e-mail address, physical address and other connection information of the customer.

paymentMethod: The type of credit card payment received from the customer (for example, payment made under Mastercard with the last four digits being 9904).

Methods:

displayCustomerInfo(): Shows the customer’s information (name, e-mail address, payment data).

**Room Class**

**Attributes:**

roomNumber: The id that is unique to the room.

roomType: The type of room (e.g. Queen Bed).

costPerNight: The price that they charge per night for the particular room.

availability: Whether the room is available or not for using it in booking.

amenities: A set or list of available services by rooms, for instance; “No Smoking,” “Desk,” “Safe,” etc.

**Methods:**

checkAvailability(): When this function is called it returns a boolean value whether there is availability of the room.

**Payment Class**

**Attributes:**

paymentID: An identification number for the particular payment that has been made.

paymentMethod: The type of payment that has been made used (as an example “Mastercard”).

amount: The overall volume of these payments, or the complete amount paid out in this regard.

billingDetails: Basic information contains north name, address, and more significantly the billing information.

**Methods:**

processPayment(): And it settles for the payment of the reservation.

**Associations:**

Reservation to Customer: A reservation is associated with a customer for the simple reason that the customer makes the reservation.

Reservation to Room: A reservation means that one get to reserve a room.

Reservation to Payment: A reservation is linked with the payment that has been made by the customer.

**3. Python** **Code**

import time

import random

import sys

def splash\_screen():

    print("--------------------------------------------------")

    print("\033[94mHOTEL RESERVATION SYSTEM\033[0m")

    print("--------------------------------------------------")

    print("Loading", end="")

    for \_ in range(6):

        time.sleep(0.5)

        print(".", end="", flush=True)

    print("\n")

def start\_or\_quit():

    while True:

        print("Options:")

        print("1. Start")

        print("2. Quit")

        choice = input("Select an option (1/2): ")

        if choice == '1':

            return True

        elif choice == '2':

            sys.exit("\nYou have exited the system. Thank you for visiting.")

        else:

            print("Invalid option. Please select again.")

class Reservation:

    def \_\_init\_\_(self, reservation\_id, name, email, phone, address, check\_in, check\_out, room, nights, room\_type, facilities):

        """

        Initializes the reservation object with user inputs.

        """

        self.reservation\_id = reservation\_id

        self.name = name

        self.email = email

        self.phone = phone

        self.address = address

        self.check\_in = check\_in

        self.check\_out = check\_out

        self.room = room

        self.nights = nights

        self.room\_type = room\_type

        self.facilities = facilities

        self.room\_cost = self.calculate\_room\_cost()

        self.facilities\_cost = self.calculate\_facilities\_cost()

        self.total\_cost = (self.room\_cost + self.facilities\_cost) \* self.nights

        self.tax = self.total\_cost \* 0.16  # 16% tax

        self.final\_total = self.total\_cost + self.tax

        self.trip\_number = random.randint(1000000000000000, 9999999999999999)

        self.confirmation\_number = random.randint(1000000, 9999999)

    def calculate\_room\_cost(self):

        """

        Calculates the cost based on the room type.

        """

        if self.room\_type == "1 Bed":

            return 50

        elif self.room\_type == "2 Queen Beds":

            return 80

        elif self.room\_type == "3 Beds":

            return 100

        else:

            return 0

    def calculate\_facilities\_cost(self):

        """

        Calculates the additional cost for selected facilities.

        """

        total = 0

        for facility in self.facilities:

            if facility == "No Smoking":

                total += 5

            elif facility == "Desk":

                total += 3

            elif facility == "Safe":

                total += 3

            elif facility == "Coffee Maker":

                total += 2

            elif facility == "Hair Dryer":

                total += 4

        return total

    def display\_reservation(self):

        """

        Displays the reservation details as per the formatted layout.

        """

        print("\033[94mYour Reservation Is Confirmed\033[0m")

        print("Thank you for your reservation. Please print your hotel receipt and show it at check-in.\n")

        print(f"Your Name: {self.name}")

        print(f"Your Email: {self.email}")

        print(f"Priceline Trip Number: {self.trip\_number}")

        print(f"Hotel Confirmation Number: {self.confirmation\_number}\n")

        print("---------------------------------------------------------------")

        print(f"Comfort Inn & Suites Los Alamos")

        print(f"Address: {self.address}")

        print(f"Check-In: {self.check\_in} - 03:00 PM")

        print(f"Check-Out: {self.check\_out} - 12:00 PM")

        print(f"Number of Nights: {self.nights}")

        print(f"Number of Rooms: 1")

        print(f"Room: {self.room} - {self.room\_type}")

        print(f"Facilities: {', '.join(self.facilities)}\n")

        print("---------------------------------------------------------------")

        print("Summary of Charges")

        print(f"Billing Name: {self.name}")

        print(f"Credit Card: Mastercard (ending in 9904)")

        print(f"Room Cost: ${self.room\_cost:.2f} avg. per room, per night")

        print(f"Rooms: 1")

        print(f"Nights: {self.nights}")

        print(f"Room Subtotal: ${self.total\_cost:.2f}")

        print(f"Taxes & Fees: ${self.tax:.2f}")

        print(f"Total Charges: ${self.final\_total:.2f} (prices are in US dollars)\n")

class Hotel:

    def \_\_init\_\_(self):

        """

        Initializes the hotel with a set of rooms and their availability.

        """

        self.rooms = {1: "Available", 2: "Available", 3: "Available"}

    def check\_room\_availability(self, room\_number):

        """

        Checks if a room is available.

        """

        return self.rooms[room\_number] == "Available"

    def book\_room(self, room\_number):

        """

        Books the room by setting it to 'Booked'.

        """

        self.rooms[room\_number] = "Booked"

    def display\_available\_rooms(self):

        """

        Displays the available rooms.

        """

        print("Available Rooms:")

        for room, status in self.rooms.items():

            if status == "Available":

                print(f"Room {room} - Available")

def reservation\_flow():

    hotel = Hotel()

    # User information taking

    name = input("Enter your name: ")

    email = input("Enter your email: ")

    phone = input("Enter your phone number: ")

    address = input("Enter your address: ")

    check\_in = input("Enter check-in date (e.g., Aug 22, 2010): ")

    check\_out = input("Enter check-out date (e.g., Aug 24, 2010): ")

    nights = int(input("Enter the number of nights: "))

    # Selecting Room

    while True:

        hotel.display\_available\_rooms()

        room\_number = int(input("Select a room number: "))

        if hotel.check\_room\_availability(room\_number):

            hotel.book\_room(room\_number)

            break

        else:

            print(f"Room {room\_number} is already booked. Please select a different room.")

    print("Room Types:")

    print("1. 1 Bed ($50 per night)")

    print("2. 2 Queen Beds ($80 per night)")

    print("3. 3 Beds ($100 per night)")

    room\_type\_choice = input("Select room type (1/2/3): ")

    if room\_type\_choice == "1":

        room\_type = "1 Bed"

    elif room\_type\_choice == "2":

        room\_type = "2 Queen Beds"

    elif room\_type\_choice == "3":

        room\_type = "3 Beds"

    else:

        room\_type = "1 Bed"

    facilities = []

    print("Available Facilities (each with additional cost):")

    print("1. No Smoking ($5)")

    print("2. Desk ($3)")

    print("3. Safe ($3)")

    print("4. Coffee Maker ($2)")

    print("5. Hair Dryer ($4)")

    while True:

        facility\_choice = input("Select a facility to add (1/2/3/4/5) or 'done' when finished: ")

        if facility\_choice == "1":

            facilities.append("No Smoking")

        elif facility\_choice == "2":

            facilities.append("Desk")

        elif facility\_choice == "3":

            facilities.append("Safe")

        elif facility\_choice == "4":

            facilities.append("Coffee Maker")

        elif facility\_choice == "5":

            facilities.append("Hair Dryer")

        elif facility\_choice.lower() == "done":

            break

        else:

            print("Invalid choice, please select again.")

    # Creating Reservation

    reservation\_id = "R123456"

    reservation = Reservation(reservation\_id, name, email, phone, address, check\_in, check\_out, room\_number, nights, room\_type, facilities)

    reservation.display\_reservation()

def main():

    splash\_screen()

    while True:

        if start\_or\_quit():

            reservation\_flow()

            while True:

                proceed = input("Do you want to make another reservation? (yes/no): ").lower()

                if proceed == 'yes':

                    break

                elif proceed == 'no':

                    print("\nThank you for using the Hotel Reservation System. Goodbye!")

                    sys.exit()

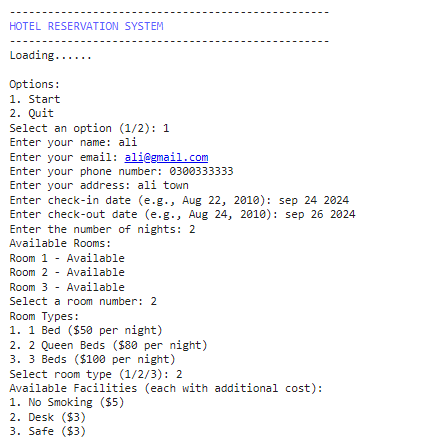
                else:

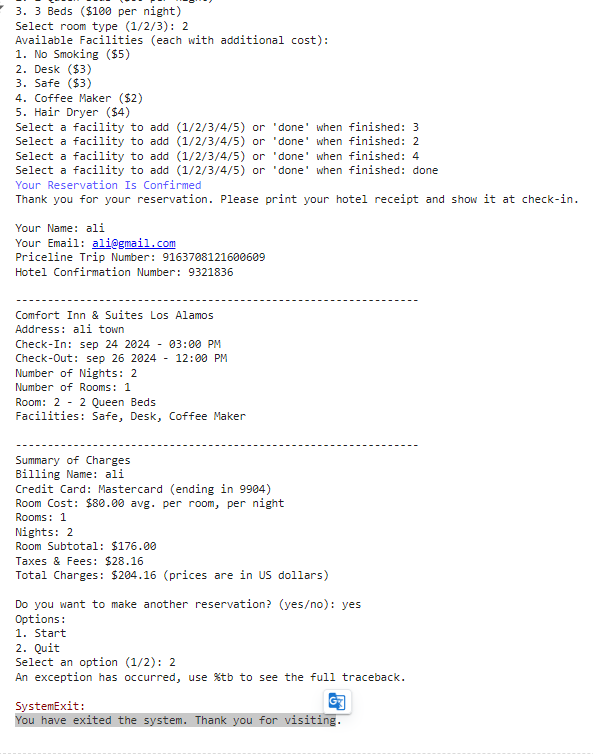
                    print("Invalid choice, please enter 'yes' or 'no'.")

if \_\_name\_\_ == "\_\_main\_\_":

    main()

**Output**

****

****

**5. Summary of learnings**

**Understanding Object-Oriented Programming (OOP):**

The assignment also helped me to gain more insight about OOP principle that included; classes, objects, constructors, attributes and methods. It was useful for me to learn how to map real-world objects (such as Reservation, Hotel, Customer) with some of their characteristics/enumerations within a class construct. The idea of dividing functionality by various classes for improved code structure and recycling was also stressed.

**User Interaction and Flow Control:**

Since the navigation through the interactive system had to have a user-like flow, creating command-line inputs was a necessity. I learned in this assignment on how to lead the user through a process of inputting their details, choosing rooms of their preference and the possible facilities for them to use. Failure to have a right control in loops and conditionals such as checking room availability, or invalid input was another lesson learnt.

**Dynamic Cost Calculations and Customizations:**

Adopting methods of computing room costs from the selected room type and additional amenities illustrated how one could dynamically compute totals. Using taxes and the rationale behind the charges is essential for facility level charges was useful to me in honing my ability to address the customizable user parameters and the costs that correspond to them.

**Randomization and Data Generation:**

The generation of random Priceline Trip Numbers and Hotel Confirmation Numbers using the Python’s random module demonstrated the essentiality of automated ID generation for unique identifiers in one real-world application.

**User Experience Enhancements:**

The incorporation of a splash screen, graphics describing a load bar, and start as well as quit options helped the students appreciate how proper interface design is crucial to developing a favorable user encounter. From this project, I realized how to enhance the way users navigate within console-based applications by giving precise headings and better utilizing of the color codes.

**Error Handling and Validation:**

Consciously making provisions for user errors including choosing a booked room or typing an incorrect value was an enlightening experience. The assignment reinstated how much necessary it is to design strong systems which afford with predicting the trajectory of user and aids him\her to approach the correct input reporting.

**Modular Code and Reusability:**

This decision also provided a more organized structure with the breakdown of the reservation process into much smaller and reusable functions. This was evident through functions such as splash\_screen(), start\_or\_quit(), and reservation\_flow(), which proved how a simple concept with a high level of integration can actually be broken down into so simple forms that the various parties involved cannot struggled to make changes to it.

**Loop Control and Program Flow Management:**

Designing a loop for restarting or exiting the reservation system was useful as it gave a basic example of how iterative systems can be constructed. This involved providing the user with an option of either continue with the reservation or get out of the program which acted as an added control on the flow of the program.

**Conclusion:**

This assignment had an all-around teaching about the enhancement of a hotel reservation system from the aspects of OOP, user connection, the dynamic calculation of fees and charges, and the improvement of the user experience. The application of these concepts in Python was helpful to transform these ideas into tangible elements when designing and developing reliable and efficient software applications that would be easily optimizable for real users.

.