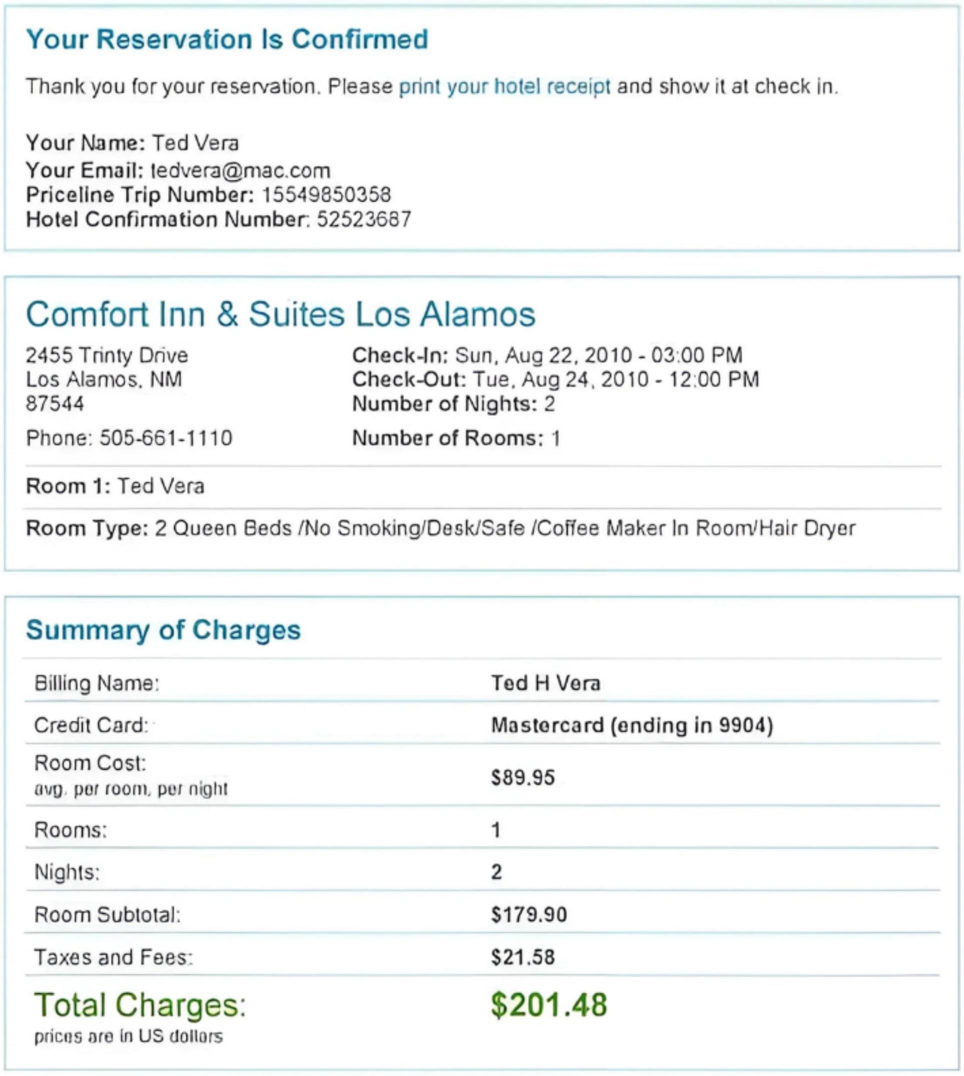
**ICS 220 – Programming Fundamentals**

**Assignment 1**

**202222015 – Ayesha Adel Almansoori**

## Description: Hotel Reservation System



**Your Reservation Is Confirmed**

Thank you for your reservation. Please print your hotel receipt and show it at check in.

**Your Name:** Ted Vera
**Your Email:** tedvera@mac.com
**Priceline Trip Number:** 15549850358
**Hotel Confirmation Number:** 52523687

**Comfort Inn & Suites Los Alamos**

| | |
|---|---|
| 2455 Trinty Drive<br>Los Alamos, NM<br>87544 | **Check-In:** Sun, Aug 22, 2010 - 03:00 PM<br>**Check-Out:** Tue, Aug 24, 2010 - 12:00 PM<br>**Number of Nights:** 2 |
| Phone: 505-661-1110 | **Number of Rooms:** 1 |

**Room 1:** Ted Vera

**Room Type:** 2 Queen Beds /No Smoking/Desk/Safe /Coffee Maker In Room/Hair Dryer

**Summary of Charges**

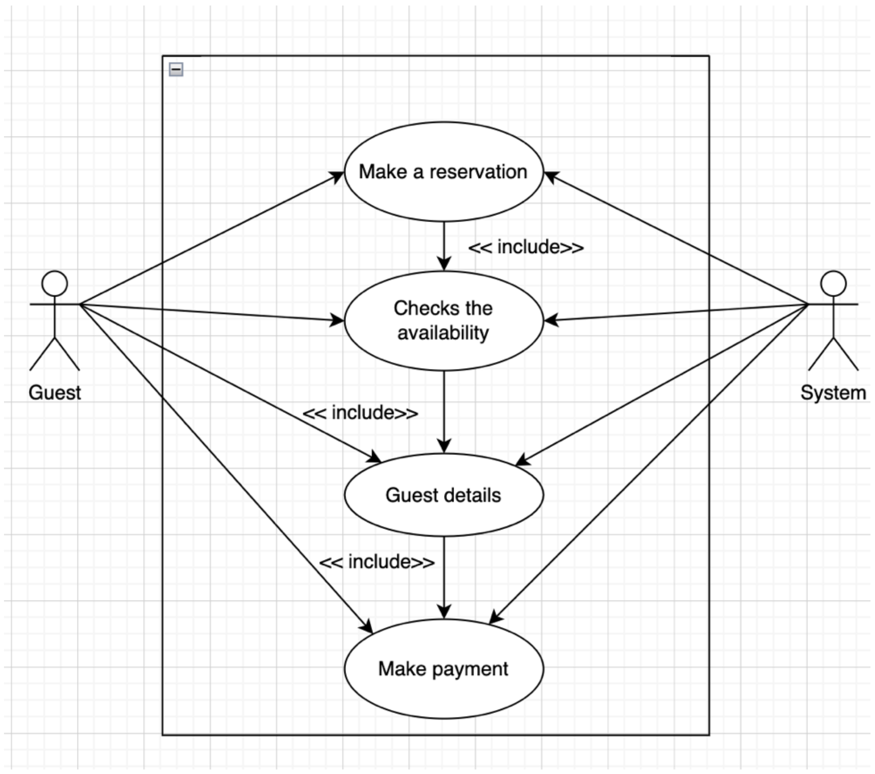
| | |
|---|---|
| Billing Name: | Ted H Vera |
| Credit Card: | Mastercard (ending in 9904) |
| Room Cost:<br>avg. per room, per night | $89.95 |
| Rooms: | 1 |
| Nights: | 2 |
| Room Subtotal: | $179.90 |
| Taxes and Fees: | $21.58 |
| Total Charges:<br>prices are in US dollars | $201.48 |

*Figure 1: Hotel Reservation Confirmation*

A hotel in Abu Dhabi requires your services to design software for managing the hotel reservation system. The above figure provides a reservation confirmation, which can be made online or at the hotel.

1) Software's UML use-case diagram and use-case description tables

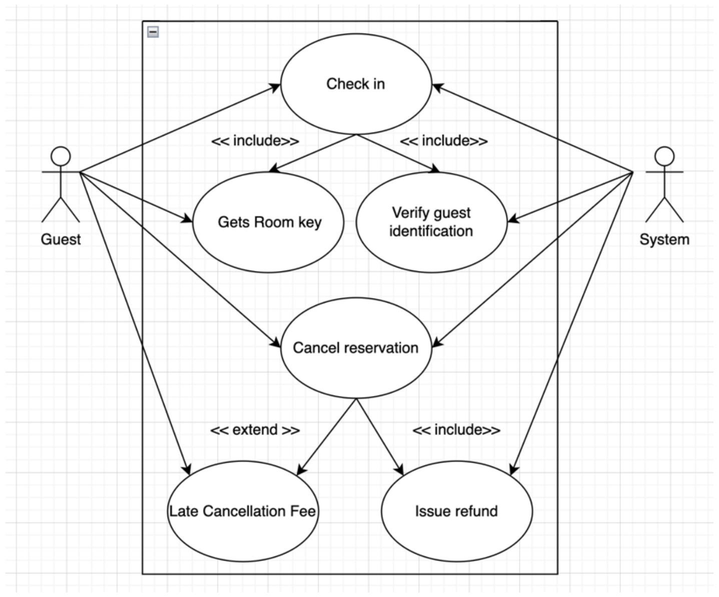
**a) <u>Scenario 1</u>**



| | |
|---|---|
| **Use - Case:** | Make a reservation |
| **Preconditions:** | Avalibality of the reserved room. |
| **Triggers:** | The guest creates a request to book a room. |
| **Main scenarito** | 1. The guest selects the rooms types and dates for staying.<br>2. The system checks if the chosen rooms are available<br>3. The guest enters reservation details and personal details<br>4. The system creates the reservation and presents the booking details.<br>5. The payment is processed successfully.<br>6. The booking is confirmed. |
| **Exceptions:** | - Insufficient balance<br>- Incorrect guest details<br>- Selected room is not avaliable<br>- Failure in the system |

| | |
|---|---|
| **Use-Case:** | Make the payment |
| **Preconditions:** | A suffcient payment must be avalible |
| **Triggers:** | The guest wants to pay |
| **Main Scenarios:** | 1. The guest requests to pay.<br>2. The system presents the amount .<br>3. The system presents the payment methods.<br>4. The guest chooses the payment method and pays.<br>5. The system process the payment<br>6. The system sends the receipt and details. |
| **Exceptions:** | - Payment failure<br>- System failure |

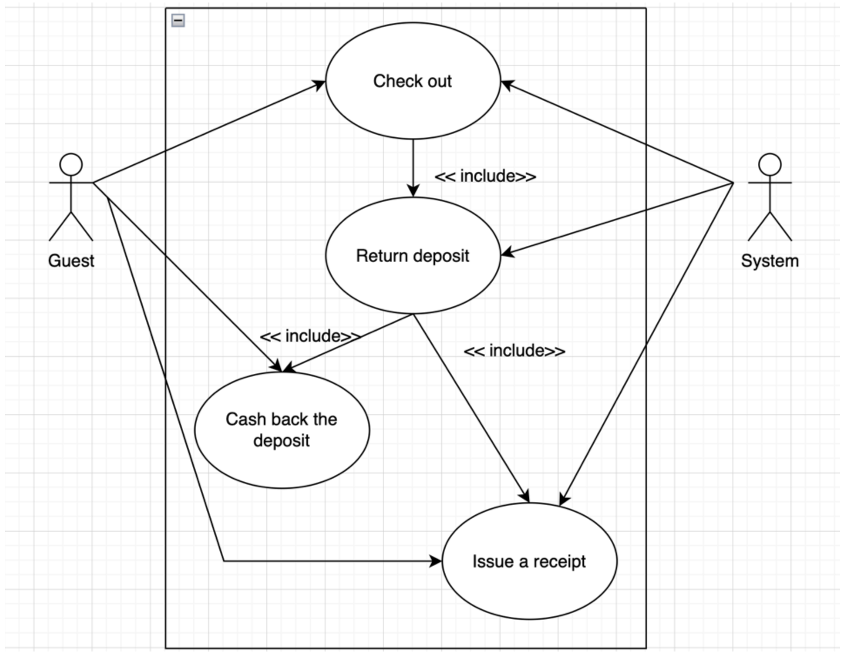
## b) Scenario 2:



| | |
|---|---|
| **Use - Case:** | Cancel a Reservation |
| **Preconditions:** | - There is an existing reservation already.<br>- Cancellation must be made in a specific period of time |
| **Triggers:** | A guest requests to cancel the reservation. |
| **Main scenarito** | 1. The guest selects the booked reservation to cancel it.<br>2. The system verifies according to the cancellation policy<br>3. The system approves the cancellation and edits the reservation status<br>4. A cancellation confirmation is sent.<br>5. Refunds are issued if applicable. |
| **Exceptions:** | - Incorrect guest details<br>- Failure in the system<br>- Reservations cannot be canceled after the deadline.<br>- Late cancellations may result in additional fees.<br>- Guests will be notified if their reservation is non-refundable. |

| | |
|---|---|
| **Use-Case:** | Check-In Guest |
| **Preconditions:** | - The guest has a valid form of reservation.<br>- The room is available.<br>- All guests must have identifications. |
| **Triggers:** | The guest requests to check-in. |
| **Main Scenarios:** | 1. The guest presents a form of reservation.<br>2. The system confirms that the reservation is valid in the system.<br>3. The guests present identifications.<br>4. The guests gives the security deposit.<br>5. The system confirms the room details , availability, and payment details.<br>6. The system updates the reservation status to checked In.<br>7. The guests recieves the check in forms and key access. |
| **Exceptions:** | - Unvalid identifications of guests the check In will be rejected<br>- Reservation is not valid in the system<br>- The paymen is not verified |

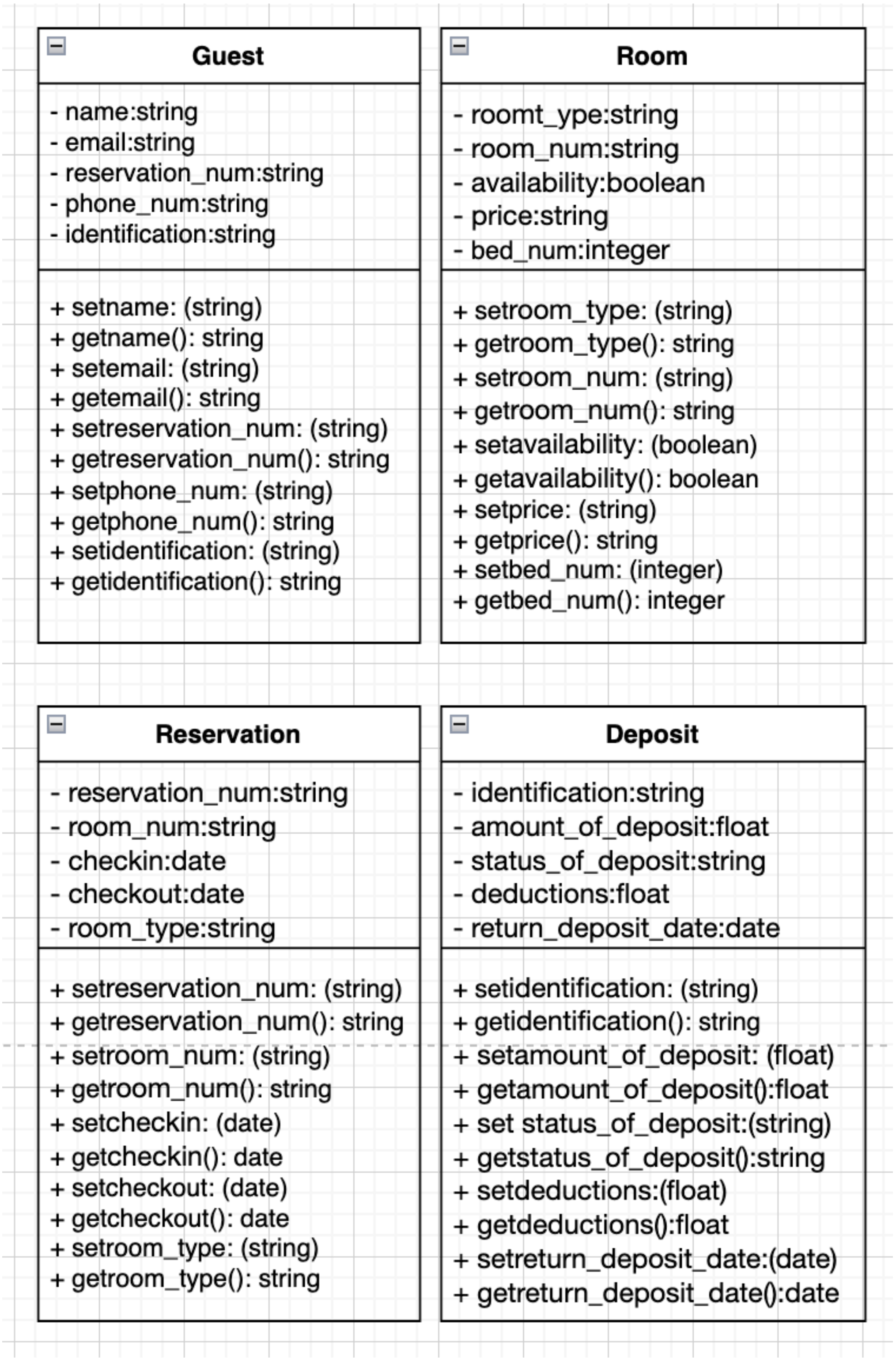
## c) **Scenario 3:**



| | |
|---|---|
| **Use-Case:** | Return Security Deposit |
| **Preconditions:** | The guest must have already checked out.<br>No unpaid charges should remain on the guest's account. |
| **Triggers:** | The guest has completed the check-out process, and the security deposit needs to be refunded. |
| **Main Scenarios:** | 1. System ensures all charges are fully paid.<br>2. The system calculates any applicable deductions from the deposit<br>3. The remaining deposit balance is refunded to the guest.<br>4. A receipt or confirmation of the returned deposit is sent to the guest. |
| **Exceptions:** | - if there are unpaid charges, the deposit is used to cover them, and only the remainder, if any, is refunded.<br>- If the payment system is down, the deposit return may be delayed or handled manually. |

| | |
|---|---|
| **Use-Case:** | Check-Out Guest |
| **Preconditions:** | The guest completed the stay |
| **Triggers:** | The guest finished the stay and wants to check out |
| **Main Scenarios:** | 1. The customer asks to check out<br>2. Room key is returned<br>3. Receptionist reviews the guest's reservation details<br>4. System creats the final bill<br>5. Customer makes any final payments<br>6. Payment is processed by the guest.<br>7. System generates a form of the stay and completes the check-out process.<br>8. Room status is updated to available. |
| **Exceptions:** | - If the guest requests a late check-out, additional fees may be applied.<br>- If payment fails, check-out cannot be completed. |

## 2) UMLC lass Diagram and Descriptions



**Python class Guest:**

```python
class Guest:
    """Class that identifies a hotel guest."""

    def __init__(self, name, email, reservation_num, phone_num,
identification):
        # Initializing 5 attributes of Guest
        self._name = name
        self._email = email
        self._reservation_num = reservation_num
        self._phone_num = phone_num
        self._identification = identification

    # Get name
    def get_name(self):
        return self._name
    # Get email
    def get_email(self):
        return self._email
    # Get reservation number
    def get_reservation_num(self):
        return self._reservation_num
    # Get phone number
    def get_phone_num(self):
        return self._phone_num
    # Get identification
    def get_identification(self):
        return self._identification

    # Set name
    def set_name(self, name):
        self._name = name
    # Set email
    def set_email(self, email):
        self._email = email
    # Set reservation number
    def set_reservation_num(self, reservation_num):
        self._reservation_num = reservation_num
    # Set phone number
    def set_phone_num(self, phone_num):
        self._phone_num = phone_num
    # Set identification
    def set_identification(self, identification):
        self._identification = identification
```

**Python class Room:**

```python
class Room:
    """Class that identifies a hotel room."""

    def __init__(self, room_type, room_num, availability, price, bed_num):
        # Initializing 5 attributes of Room
        self._room_type = room_type
        self._room_num = room_num
        self._availability = availability
        self._price = price
        self._bed_num = bed_num

    # Get room type
    def get_room_type(self):
        return self._room_type
    # Get room number
    def get_room_num(self):
        return self._room_num
    # Get room availability
    def get_availability(self):
        return self._availability
    # Get room price
    def get_price(self):
        return self._price
    # Get bed number
    def get_bed_num(self):
        return self._bed_num

    # Set room type
    def set_room_type(self, room_type):
        self._room_type = room_type
    # Set room number
    def set_room_num(self, room_num):
        self._room_num = room_num
    # Set room availability
    def set_availability(self, availability):
        self._availability = availability
    # Set room price
    def set_price(self, price):
        self._price = price
    # Set bed number
    def set_bed_num(self, bed_num):
        self._bed_num = bed_num
```

**Python class Reservation:**

```python
class Reservation:
    """Class that identifies a reservation made by a guest."""

    def __init__(self, reservation_num, room_num, checkin, checkout,
room_type):
        # Initializing 5 attributes of Reservation
        self._reservation_num = reservation_num
        self._room_num = room_num
        self._checkin = checkin
        self._checkout = checkout
        self._room_type = room_type

    # Get reservation number
    def get_reservation_num(self):
        return self._reservation_num
    # Get room number
    def get_room_num(self):
        return self._room_num
    # Get check in
    def get_checkin(self):
        return self._checkin
    # Get check out
    def get_checkout(self):
        return self._checkout
    # Get room type
    def get_room_type(self):
        return self._room_type

    # Set reservation number
    def set_reservation_num(self, reservation_num):
        self._reservation_num = reservation_num
    # Set room number
    def set_room_num(self, room_num):
        self._room_num = room_num
    # Set check in
    def set_checkin(self, checkin):
        self._checkin = checkin
    # Set check out
    def set_checkout(self, checkout):
        self._checkout = checkout
    # Set room type
    def set_room_type(self, room_type):
        self._room_type = room_type
```

**Python class Deposit:**

```python
class Deposit:
    """Class that identifies a deposit for a reservation."""

    def __init__(self, identification, amount_of_deposit, status_of_deposit,
deductions, return_deposit_date):
        # Initializing 5 attributes of Deposit
        self._identification = identification
        self._amount_of_deposit = amount_of_deposit
        self._status_of_deposit = status_of_deposit
        self._deductions = deductions
        self._return_deposit_date = return_deposit_date

    # Get identification
    def get_identification(self):
        return self._identification
    #Get amount of deposit
    def get_amount_of_deposit(self):
        return self._amount_of_deposit
    #Get status of deposit
    def get_status_of_deposit(self):
        return self._status_of_deposit
    #Get deductions
    def get_deductions(self):
        return self._deductions
    #Get return deposit date
    def get_return_deposit_date(self):
        return self._return_deposit_date


    #Set identification
    def set_identification(self, identification):
        self._identification = identification
    #Set amount of deposit
    def set_amount_of_deposit(self, amount_of_deposit):
        self._amount_of_deposit = amount_of_deposit
    #Set status of deposit
    def set_status_of_deposit(self, status_of_deposit):
        self._status_of_deposit = status_of_deposit
    #Set deductions
    def set_deductions(self, deductions):
        self._deductions = deductions
    # Set return deposit date
    def set_return_deposit_date(self, return_deposit_date):
        self._return_deposit_date = return_deposit_date
```

**The objects of all classes**

```
    # Creating an object for the Guest , Room , Reservation, Deposit class.
guest1 = Guest(name="Ayesha Adel", email="ayesha.adel@gmail.com",
reservation_num="R2457", phone_num="0555588242",
               identification="ID7842004")

room1 = Room(room_type="Deluxe", room_num="111", availability=True,
price="200.00", bed_num=2)

reservation1 = Reservation(reservation_num="R2457", room_num="111",
checkin="2024-10-1", checkout="2024-10-2",
                           room_type="Deluxe")

deposit1 = Deposit(identification="ID7842004", amount_of_deposit=500.00,
status_of_deposit="Confirmed",
                   deductions=50.00, return_deposit_date="2024-10-2")
```

5. **GitHub repository link:**

   https://github.com/Ayesha-am/Ass1.git

**5. Summary of learnings**

In this assignment, I completed three main parts. In Part 1, I created a UML use-case diagram along with detailed use-case descriptions for the hotel reservation system. These descriptions included preconditions, triggers, main scenarios, and exceptions for various processes such as making a reservation, making a payment, canceling a reservation, checking in, checking out guests, and security deposit. In Part 2, I designed UML class diagrams and implemented Python classes representing the main elements of the reservation system, such as the guest, room, reservation, and deposit. Each class had attributes and methods to manage the necessary data and functionality, covering getters and setters for key properties. In Part 3, I brought all the classes together to create the hotel reservation process, creating objects for each class and how they interact. Through this assignment, I learned how to create a real-world system using UML diagrams and effectively tranfrom those models into Python classes, emphasizing the importance of object-oriented principles such as data hiding and abstraction in software systems.