Northeastern University
Khoury College of Computer Sciences

DS 5110 : Introduction to Data Management Processing

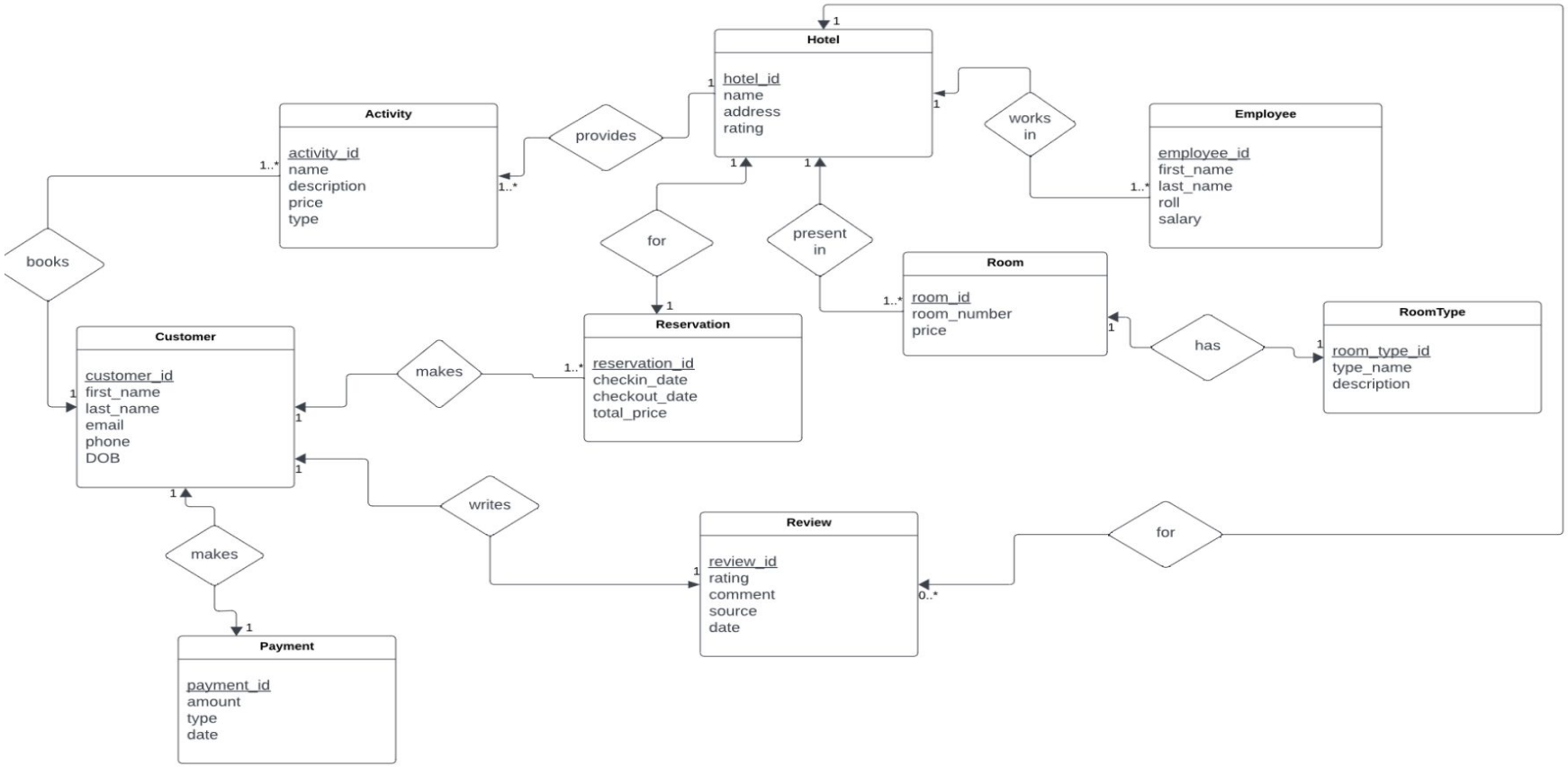
HOTEL MANAGEMENT SYSTEM

Krithika Iyer
Rahul Chettri
Rohan Shriniwas Devasthale

Introduction

- Hotel Reservation and Management Database System that allows to store information about hotels, rooms, activities, customer details and booking information.
- Use of Relational Database (SQL) to store data and create a booking application using python that allows user to login and book rooms and activities.
- In addition to booking also allows customer to add reviews.
- System provides two roles - customer and admin, admin role allows to add payment information, view analytics and update salaries of the employees.

Database Design (ER Diagram)



DATABASE TABLES

- customer (customer_id , first_name, last_name, email, phone, DOB);
- hotel (hotel_id, name, address, rating);
- room (room_id, room_number, room_type_id, hotel_id, price, hotel(hotel_id),roomType(room_type_id));
- roomType (room_type_id, type_name, description);
- reservation (reservation_id, customer_id, hotel_id, checkin_date, checkout_date, guest_count, total_price customer(customer_id), hotel(hotel_id));
- reservationDetails (reservation_id, room_id, guest_count, reservation(reservation_id), room(room_id));

DATABASE TABLES CONTINUED

- activity (activity_id, name, description, price,type)
- activityBooking (booking_id, activity_id, customer_id, date, activity(activity_id), customer(customer_id));
- employee (employee_id, name, role, DOB, salary, hotel(hotel_id));
- payment (payment_id, customer_id, amount, type, date, customer(customer_id));
- review (review_id, customer_id, hotel_id, rating, comment, source, customer(customer_id), hotel(hotel_id));
- roomAvailability (availability_id, room_id, hotel_id, room_type_id, is_available, date, cleaning_status, room(room_id));

STORED PROCEDURES (UPDATE CUSTOMER)

```
-- procedure to update customer details
DELIMITER //

CREATE PROCEDURE updateCustomerDetails(
    IN customerId INT,
    IN newPhoneNumber VARCHAR(15),
    IN newEmail VARCHAR(255)
)
BEGIN
    UPDATE customer
    SET phone = newPhoneNumber, email = newEmail
    WHERE customer_id = customerId;
END //

DELIMITER ;

CALL updateCustomerDetails(1002, '555-0301', 'vivaan@gmail.com');
```

stored procedure to update customer information

BEFORE

1002	Vivaan	Mehta	vivaan.mehta@gmail.com	555-0202	1990-07-19

AFTER

1002	Vivaan	Mehta	vivaan@gmail.com	555-0301	1990-07-19

STORED PROCEDURES (VIEW RESERVATION)

```
CREATE PROCEDURE GetCustomerReservationDetails(IN customerID INT)
BEGIN
    SELECT
        c.first_name,
        c.last_name,
        h.name AS hotel_name,
        h.address AS hotel_address,
        rt.type_name AS room_type,
        r.room_number,
        res.checkin_date,
        res.checkout_date
    FROM
        customer c
    left JOIN reservation res ON c.customer_id = res.customer_id
    JOIN reservationDetails rd ON res.reservation_id = rd.reservation_id
    JOIN room r ON rd.room_id = r.room_id
    JOIN roomType rt ON r.room_type_id = rt.room_type_id
    JOIN hotel h ON r.hotel_id = h.hotel_id
    WHERE
        c.customer_id = customerID;
END //

DELIMITER ;
select * from customer;
CALL GetCustomerReservationDetails(1001);
```

USE OF THREE JOINS

STORED PROCEDURES (VIEW RESERVATION)

OUTPUT

```
3   CALL GetCustomerReservationDetails(1001);
4
```

00% 1:4

Result Grid Filter Rows: Search Export:

first_name	last_name	hotel_name	hotel_address	room_type	room_number	checkin_da...	checkout_da...
Aarav	Kumar	The Golden Gate Hotel	100 Bridge Plaza, San Francisco, CA	Standard	101	2023-12-01	2023-12-02

STORED PROCEDURES (BOOK ROOM PT 1.)

```
DELIMITER //

CREATE PROCEDURE GetAvailableRoom(
    IN in_hotel_id INT,
    IN in_room_type_id INT,
    IN in_checkin_date DATE,
    IN in_checkout_date DATE,
    OUT next_room_id INT
)
BEGIN
    SELECT MIN(ra.room_id) INTO next_room_id
    FROM roomAvailability ra
    WHERE ra.room_type_id = in_room_type_id
    AND ra.hotel_id = in_hotel_id
    AND ra.date BETWEEN in_checkin_date AND in_checkout_date
    AND ra.is_available = 1;

    IF next_room_id IS NULL THEN
        SET next_room_id = NULL;
    END IF;
END //

DELIMITER ;
```

stored procedure accepts the hotel id which customer wants to book, room type, check-in date, check-out date and returns the available room_id that meets the criteria.

STORED PROCEDURES (BOOK ROOM PT2.)

```mysql
-- Reserve Room assuming each extra guest costs same

DELIMITER //

CREATE PROCEDURE ReserveRoom(
    IN in_room_id INT,
    IN in_customer_id INT,
    IN in_checkin_date DATE,
    IN in_checkout_date DATE,
    IN in_guest_count INT,
    OUT out_reservation_id INT
)
BEGIN

    DECLARE is_room_available INT;
    DECLARE booking_hotel_id INT;

    SELECT COUNT(*) INTO is_room_available
    FROM roomAvailability ra
    WHERE ra.room_id = in_room_id
        AND ra.date BETWEEN in_checkin_date AND in_checkout_date
        AND ra.is_available = 1;
    SELECT hotel_id INTO booking_hotel_id FROM room WHERE room_id = in_room_id LIMIT 1;

    IF is_room_available >= DATEDIFF(in_checkin_date, in_checkout_date) + 1 THEN
        INSERT INTO reservation (customer_id, hotel_id, checkin_date, checkout_date, guest_count, total_price)
        VALUES (in_customer_id, booking_hotel_id, in_checkin_date, in_checkout_date, in_guest_count,
        CALCULATE_TOTAL_PRICE(in_checkin_date, in_checkout_date, in_room_id, in_guest_count));

        SET out_reservation_id = LAST_INSERT_ID();

        INSERT INTO reservationDetails (reservation_id, room_id, guest_count)
        VALUES (out_reservation_id, in_room_id, in_guest_count);
    ELSE
        SET out_reservation_id = NULL;
    END IF;
END //

DELIMITER ;
```

- books a room for given dates and customer info.
- Returns reservationid.
- This procedure first checks if given room_id is available, if so inserts details into reservation & reservationDetails table.
- It also calculates total cost using function and updates the status of room at roomAvailability Table using trigger.

STORED PROCEDURES (BOOK ROOM)

```
-- Book Room Procedure
DELIMITER //

CREATE PROCEDURE BookRoom(
    IN customer_id INT,
    IN guest_count INT,
    IN hotel_id INT,
    IN room_type_id INT,
    IN checkin_date DATE,
    IN checkout_date DATE
)
BEGIN
    DECLARE available_room_id INT;
    DECLARE reserved_room_id INT;

    CALL GetAvailableRoom(hotel_id, room_type_id, checkin_date, checkout_date, available_room_id);

    CALL ReserveRoom(available_room_id, customer_id, checkin_date, checkout_date, guest_count, reserved_room_id);
END //

DELIMITER ;
```

a wrapper stored procedure that takes booking information (customer details, hotel details, check-in, check-out dates and guest info). first checks available room for given input using getAvailableRoom and then books the room using reserveRoom stored proc.

TRIGGER

```
DELIMITER //

CREATE TRIGGER UpdateRoomAvailabilityAfterReservation
AFTER INSERT ON reservationDetails
FOR EACH ROW
BEGIN
    DECLARE checkindate DATE;
    DECLARE checkoutdate DATE;

    SELECT checkin_date, checkout_date INTO checkindate, checkoutdate
    FROM reservation
    WHERE reservation_id = NEW.reservation_id;

    UPDATE roomAvailability
    SET is_available = 0
    WHERE room_id = NEW.room_id
        AND date BETWEEN checkindate AND checkoutdate;
END //

DELIMITER ;
```

AFTER INSERT TRIGGER ON RESERVATIONDETAILS TABLE, Once a room is booked using the bookRoom stored procedure, this trigger ensures that the status of the booked room is unavailable for the booked dates. Updates status on roomAvailability Table

Functions

```
-- Function to calculate the total price for the customer

DELIMITER //
CREATE FUNCTION CALCULATE_TOTAL_PRICE(
    checkin_date_param DATE,
    checkout_date_param DATE,
    room_id_param INT,
    guest_count_param INT
)
RETURNS DECIMAL(10, 2) DETERMINISTIC
BEGIN
    DECLARE total_price DECIMAL(10, 2);

    SELECT price INTO total_price
    FROM room
    WHERE room_id = room_id_param;

    SET total_price = total_price * (DATEDIFF(checkout_date_param, checkin_date_param) + 1);

    RETURN total_price;
END //
DELIMITER ;
```

CALCULATE_TOTAL_PRICE

Purpose: This function calculates the total price for a room booking.

Inputs:

- **checkin_date_param -** The date when the guest checks in.
- **checkout_date_param -** The date when the guest checks out.
- **room_id_param -** The identifier of the room being booked.
- **guest_count_param -**The number of guests (note: this parameter is not actually used in the calculation).

Functions

```sql
-- Function to calculate the total price for the customer during checkout (includes activity price)
DELIMITER //

CREATE FUNCTION CALCULATE_TOTAL_BILL(
    in_customer_id INT,
    in_checkin_date DATE,
    in_checkout_date DATE
) RETURNS DECIMAL(10, 2) READS SQL DATA
BEGIN
    DECLARE total_reservation_amount DECIMAL(10, 2);
    DECLARE total_activity_amount DECIMAL(10, 2);
    DECLARE total_bill DECIMAL(10, 2);

    SELECT total_price
    INTO total_reservation_amount
    FROM reservation
    WHERE customer_id = in_customer_id
      AND checkin_date = in_checkin_date
      AND checkout_date = in_checkout_date;

    SELECT COALESCE(SUM(a.price), 0)
    INTO total_activity_amount
    FROM activityBooking ab
    JOIN activity a ON ab.activity_id = a.activity_id
    WHERE ab.customer_id = in_customer_id
      AND ab.date >= in_checkin_date
      AND ab.date <= in_checkout_date;

    SET total_bill = total_reservation_amount + total_activity_amount;

    RETURN total_bill;
END //

DELIMITER ;
```

CALCULATE_TOTAL_BILL

Purpose: This function calculates the total bill for a customer including their room reservation and any activities they booked.

Inputs:

- **in_customer_id** - The identifier of the customer.
- **in_checkin_date** and **in_checkout_date** - The dates defining the customer's stay period.

Functions

```
-- Function to increment salary for the employee
DELIMITER //

CREATE FUNCTION INCREMENT_SALARY(
    in_employee_id INT
) RETURNS DECIMAL(10, 2) READS SQL DATA
BEGIN
    SET @original_salary := (SELECT salary FROM employee WHERE employee_id = in_employee_id);

    IF @original_salary IS NOT NULL THEN
        -- 10% increment
        SET @new_salary := @original_salary * 1.10;

        UPDATE employee
        SET salary = @new_salary
        WHERE employee_id = in_employee_id;

        RETURN @new_salary;
    ELSE
        RETURN NULL;
    END IF;
END //

DELIMITER ;
```

INCREMENT_SALARY

- **Purpose**: This function increases an employee's salary by 10%.
- **Inputs**:
  - **in_employee_id** - The identifier of the employee whose salary is to be incremented.

VIEWS

```sql
-- View for highest rated hotels

CREATE VIEW highestRatedHotels AS
SELECT
    h.name AS hotel_name,
    AVG(r.rating) AS average_rating
FROM
    hotel h
JOIN
    review r ON h.hotel_id = r.hotel_id
GROUP BY
    h.hotel_id
ORDER BY
    average_rating;
```

highestRatedHotels

Purpose: This view displays a list of hotels with their average ratings.

Columns:

- **hotel_name:** The name of the hotel.
- **average_rating:** The average rating of the hotel based on customer reviews.

VIEWS

```sql
-- View for most booked activities

CREATE VIEW mostBookedActivities AS
SELECT
    a.name AS activity_name,
    COUNT(ab.booking_id) AS times_booked
FROM
    activity a
JOIN
    activityBooking ab ON a.activity_id = ab.activity_id
GROUP BY
    a.activity_id
ORDER BY
times_booked DESC;
```

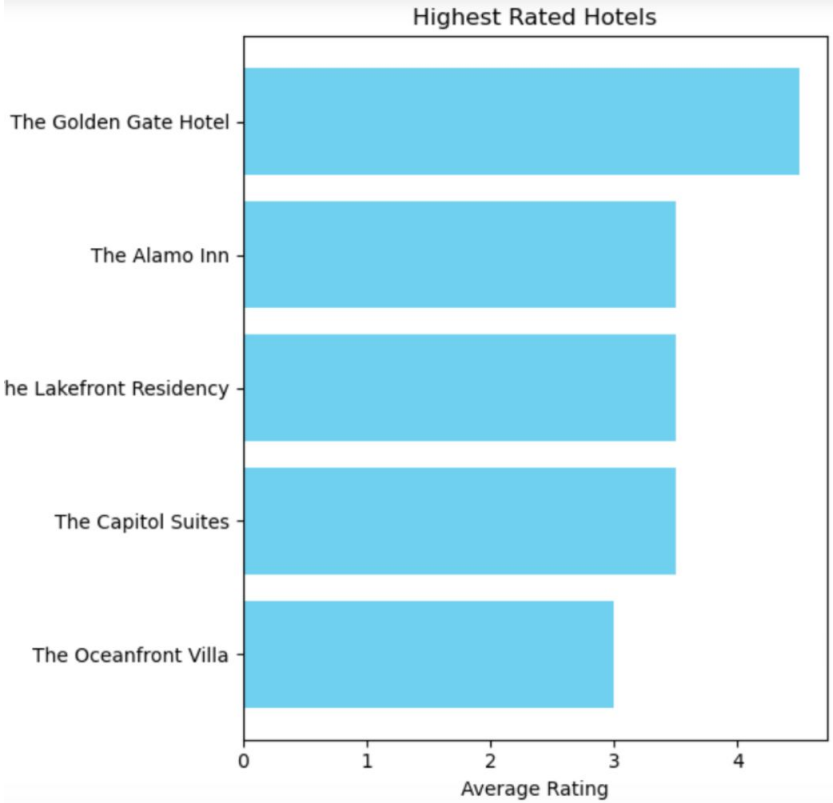
mostBookedActivities

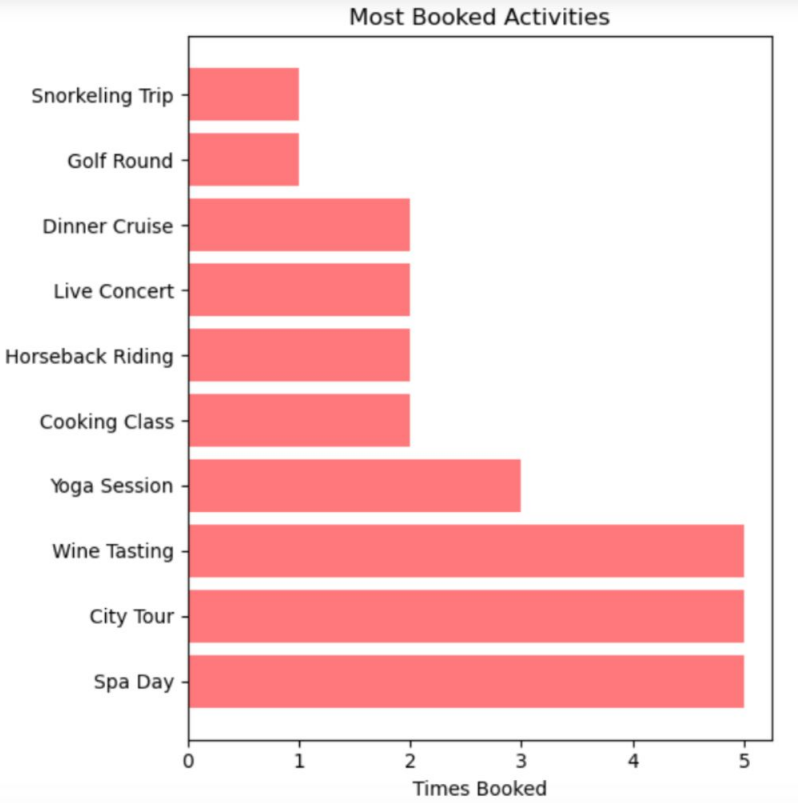
Purpose: This view shows a list of activities and how often they have been booked.

Columns:

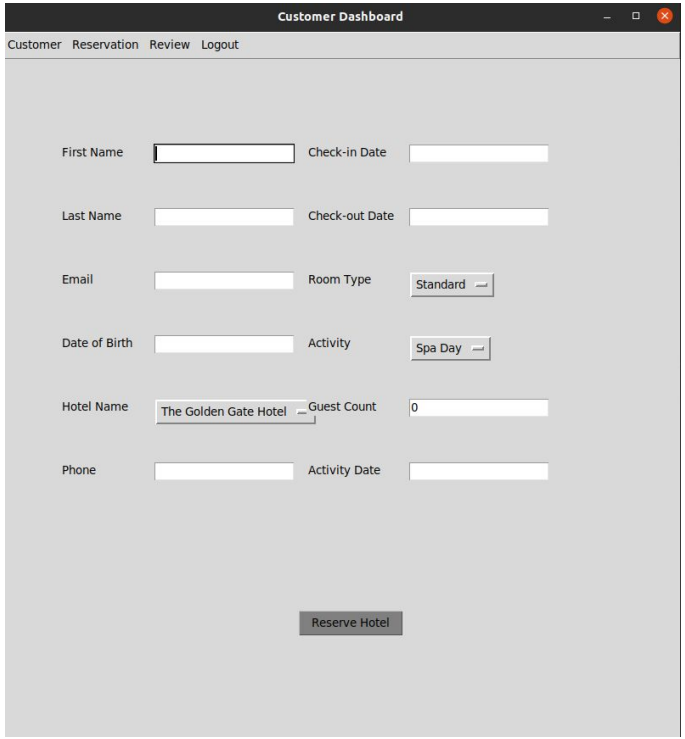
- **activity_name:** The name of the activity.
- **times_booked:** The number of times the activity has been booked.

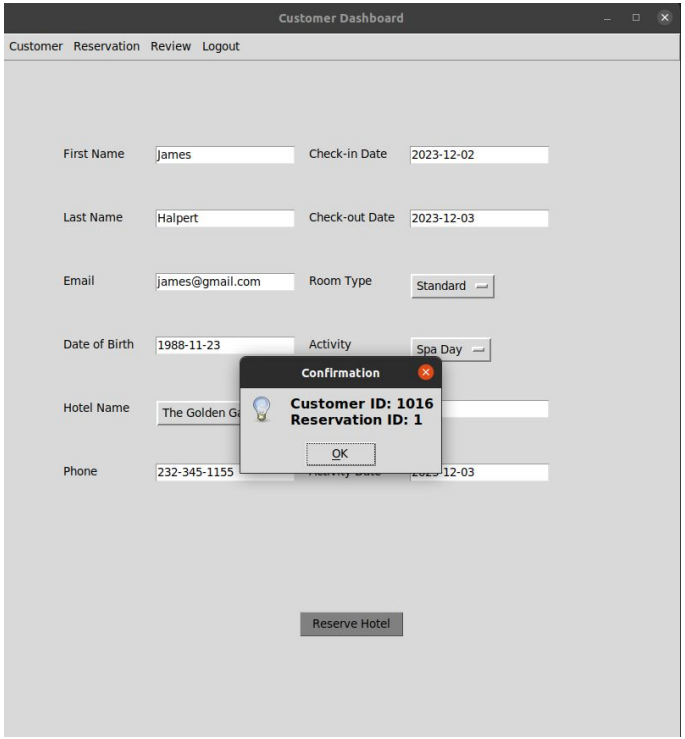
ANALYSIS


Highest Rated Hotels
The Golden Gate Hotel
The Alamo Inn
he Lakefront Residency
The Capitol Suites
The Oceanfront Villa
0
1
2
3
4
Average Rating

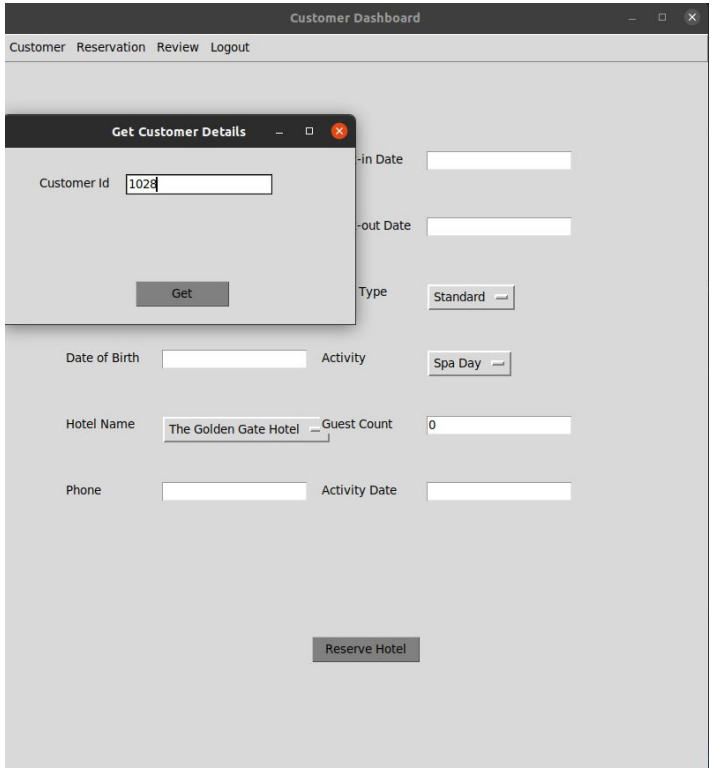

Most Booked Activities
Snorkeling Trip
Golf Round
Dinner Cruise
Live Concert
Horseback Riding
Cooking Class
Yoga Session
Wine Tasting
City Tour
Spa Day
0
1
2
3
4
5
Times Booked

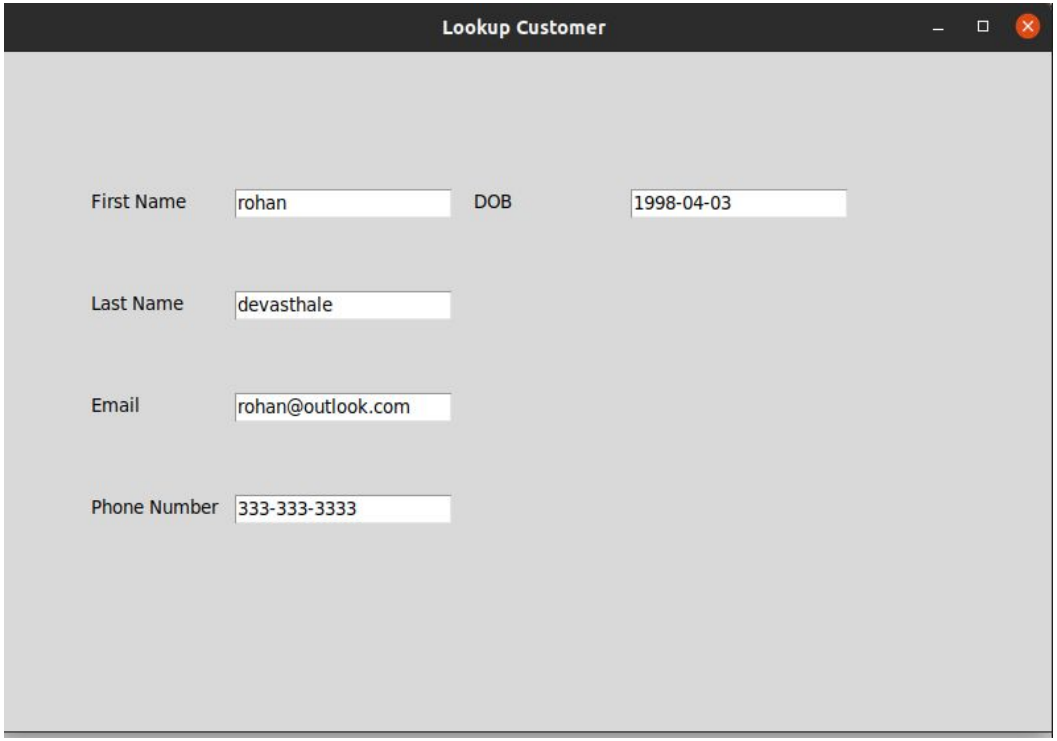
CUSTOMER FEATURES



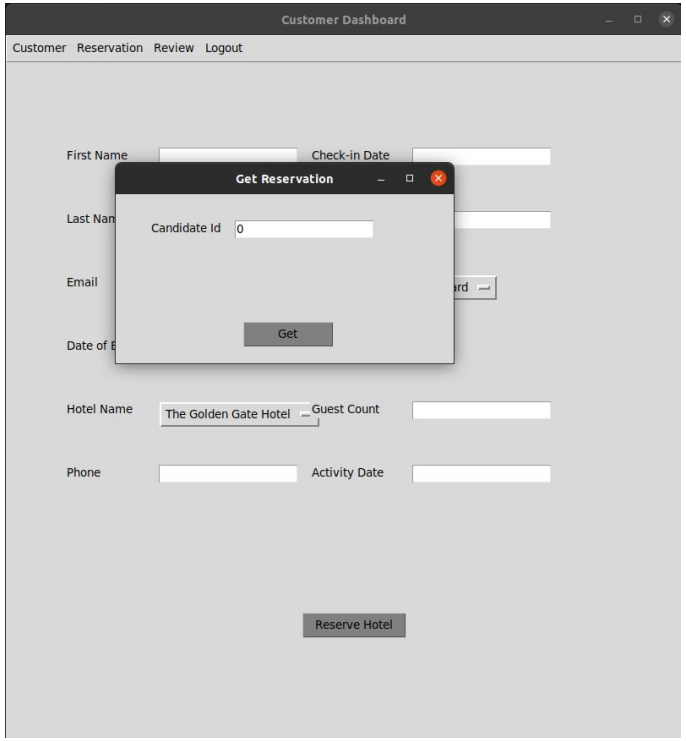


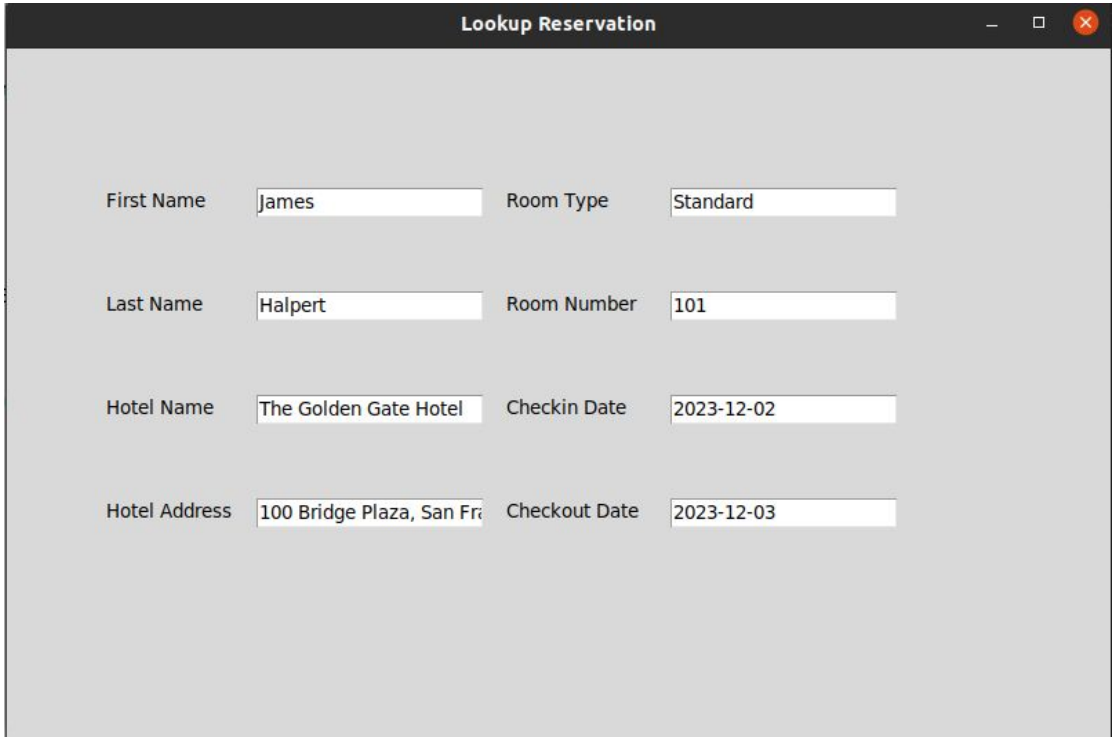
CUSTOMER FEATURES CONTINUED





CUSTOMER FEATURES CONTINUED





MORE CUSTOMER FEATURES

- Update Account
- Delete Account
- Update Reservation
- Delete Reservation
- Write a review
- Logout

ADMIN FEATURES

```
PROBLEMS    OUTPUT    TERMINAL    PORTS    DEBUG CONSOLE                bash - src
Are you a customer or admin: ?
Enter c for customer, a for admin or e to exit: a
1. Get Analysis
2. Get Final Bill for Customer
3. View Employees
4. Increment Employee Salary
5. Exit
Enter the digit for operation you want to perform: 1
Successfully connected to the database
1. Get Analysis
2. Get Final Bill for Customer
3. View Employees
4. Increment Employee Salary
5. Exit
Enter the digit for operation you want to perform: 2
Enter the customer id: 1016
Enter checkin date: 2023-12-02
Enter checkout date: 2023-12-03
The total bill for customer is:  320.00
Enter the payment type (credit card / cash): credit card
1. Get Analysis
2. Get Final Bill for Customer
3. View Employees
4. Increment Employee Salary
5. Exit
Enter the digit for operation you want to perform: 3
The employee names are:
 ['John Doe', 'Jane Smith', 'Bob Brown', 'Alice Johnson', 'Steve Davis', 'Mary Wilson', 'Chris Martinez', 'Patricia Garcia', 'Michael Lee'
, 'Linda Thompson']
1. Get Analysis
2. Get Final Bill for Customer
3. View Employees
4. Increment Employee Salary
5. Exit
Enter the digit for operation you want to perform: 4
Enter the Employee Id: 5001
The updated Salary is:  30800.00
```

- Get Analysis
- Generate Final Bill for customer
- View Employees
- Increment Salary of Employee

CONCLUSION AND FUTURE SCOPE

- We implemented a simple and user-friendly Hotel Management System that supports and integrates various hotel operations to streamline the process and enhance customer experience.
- The system involves complex stored procedures, triggers, functions and views which work at the backend seamlessly along with partial support for UI and command line throughout the application.
- In future, more operations can be supported by the admins along with a UI.
- The data generated can be analysed to enhance the system based on customer reservations, activity bookings and reviews.
- The system can be enhanced to support restaurant, cafe and dedicated workspaces.

THANK YOU

Source code: Hotel Reservation System