```
import java.sql.*;
import java.util.Scanner;
class JdbcJava {
   private static final String url = "jdbc:mysql://localhost:3306/hotel_database";
   private static final String un = "root";
   private static final String pwd = "root";

   public static void main(String[] args) {
       System.out.println("+---------------------------------------------------+");
       System.out.println("+🙏HELLO USER WELCOME TO MY HOTEL MANAGEMENT
SYSTEM🙏+");
       System.out.println("+---------------------------------------------------+");

       try {
           Class.forName("com.mysql.cj.jdbc.Driver");
       } catch (ClassNotFoundException e) {
           e.printStackTrace();
           return;
       }

       try (Connection connection = DriverManager.getConnection(url,un,pwd);
            Scanner scanner = new Scanner(System.in)) {

           while (true) {
               System.out.println();
               System.out.println("+--------------------------+");
               System.out.println("Please enter any option: ");
               System.out.println("Press 1 to Reserve a room");
               System.out.println("Press 2 to View Reservations");
               System.out.println("Press 3 to Get Room Number");
               System.out.println("press 4 to Update Reservations");
               System.out.println("Press 5 to Delete Reservations");
               System.out.println("Press 6 to Exit from the System");
               System.out.println("+--------------------------+");
               int choice = scanner.nextInt();
               scanner.nextLine();
               switch (choice) {
                   case 1:
                       reserveRoom(connection, scanner);
                       break;
                   case 2:
                       viewReservations(connection);
                       break;
                   case 3:
                       getRoomNumber(connection, scanner);
                       break;
                   case 4:
                       updateReservation(connection, scanner);
                       break;
                   case 5:
                       deleteReservation(connection,scanner);
                       break;
                   case 6:
                       exit();
                      return;
                   default:
                       System.out.println("Invalid choice!!!!. Please Enter Appropriate Option.");
               }
           }

       }
       catch (SQLException e)
       {
           e.printStackTrace();
       }
       catch (InterruptedException e)
       {
           e.printStackTrace();
       }
   }

   private static void reserveRoom(Connection connection, Scanner scanner) {
       System.out.print("Enter guest id: ");
       int reservationId = scanner.nextInt();
       scanner.nextLine();
       System.out.print("Enter guest name: ");
       String guestName = scanner.nextLine();
       System.out.print("Enter room number: ");
       int roomNumber = scanner.nextInt();
       scanner.nextLine();
       System.out.print("Enter contact number: ");
       String contactNumber = scanner.nextLine();
       System.out.print("Enter reservation date (YYYY-MM-DD): ");
       String reservationDate = scanner.nextLine();

       String sql = "INSERT INTO booking_details (reservation_id, guest_name, room_number,
contact_number, reservation_date) VALUES (?, ?, ?, ?, ?)";

       try (PreparedStatement preparedStatement = connection.prepareStatement(sql)) {
           preparedStatement.setInt(1, reservationId);
           preparedStatement.setString(2, guestName);
           preparedStatement.setInt(3, roomNumber);
```

```
            preparedStatement.setString(4, contactNumber);
            preparedStatement.setDate(5, Date.valueOf(reservationDate));

            int rows = preparedStatement.executeUpdate();
            if (rows > 0) {
                System.out.println("Reservation successful!");
            } else {
                System.out.println("Reservation failed.");
            }
        } catch (SQLException e) {
            System.out.println("Error while reserving room.");
            e.printStackTrace();
        }
    }

    private static void viewReservations(Connection connection) {
        String sql = "SELECT reservation_id, guest_name, room_number, contact_number,
reservation_date FROM booking_details";
        try (Statement statement = connection.createStatement();
             ResultSet resultSet = statement.executeQuery(sql)) {
            System.out.println("Current Reservations:");
            System.out.println("+----------------+-----------------+---------------+----------------------
+-------------------------+");
            System.out.println("| Reservation ID | Guest           | Room Number   | Contact Number      |
Reservation Date        |");
            System.out.println("+----------------+-----------------+---------------+----------------------
+-------------------------+");
            while (resultSet.next()) {
                int reservationId = resultSet.getInt("reservation_id");
                String guestName = resultSet.getString("guest_name");
                int roomNumber = resultSet.getInt("room_number");
                String contactNumber = resultSet.getString("contact_number");
                Date reservationDate = resultSet.getDate("reservation_date");

                System.out.printf("| %-14d | %-15s | %-13d | %-20s | %-24s|\n",
                        reservationId, guestName, roomNumber, contactNumber, reservationDate.toString());
            }
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
    private static void getRoomNumber(Connection connection, Scanner scanner) {
        System.out.print("Enter reservation ID: ");
        int reservationId = scanner.nextInt();
        scanner.nextLine();
        System.out.print("Enter guest name: ");
    String guestName = scanner.nextLine();
    String sql = "SELECT room_number FROM booking_details WHERE reservation_id = ? AND
guest_name = ?";
    try (PreparedStatement preparedStatement = connection.prepareStatement(sql)) {
      preparedStatement.setInt(1, reservationId);
      preparedStatement.setString(2, guestName);
      try (ResultSet resultSet = preparedStatement.executeQuery()) {
        if (resultSet.next()) {
          int roomNumber = resultSet.getInt("room_number");

System.out.println("+-------------------------------------------------------------------------------------------------------
----------+");
          System.out.println("The room " + roomNumber + " has already been booked by " +
guestName + " and reservation ID is " + reservationId);

System.out.println("+-------------------------------------------------------------------------------------------------------
----------+");
        } else {
          System.out.println("+-------------------------------------------------------+");
          System.out.println("|Reservation not found for the given ID and guest name. | ");
          System.out.println("+-------------------------------------------------------+");
        }
      }
    } catch (SQLException e) {
      System.out.println("Error while getting room number.");
      e.printStackTrace();
    }
  }
  private static void updateReservation(Connection connection, Scanner scanner) {
    System.out.print("Enter reservation ID to update: ");
    int reservationId = scanner.nextInt();
    scanner.nextLine();
    if (!reservationExists(connection, reservationId)) {
      System.out.println("Reservation not found for the given ID.");
      return;
    }
    System.out.print("Enter new guest name: ");
    String newGuestName = scanner.nextLine();
    System.out.print("Enter new room number: ");
    int newRoomNumber = scanner.nextInt();
    scanner.nextLine();
    System.out.print("Enter new contact number: ");
    String newContactNumber = scanner.nextLine();
    System.out.print("Enter new reservation date (YYYY-MM-DD): ");
    String newReservationDate = scanner.nextLine();
    String sql = "UPDATE booking_details SET guest_name = ?, room_number = ?,
contact_number = ?, reservation_date = ? WHERE reservation_id = ?";
    try (PreparedStatement preparedStatement = connection.prepareStatement(sql)) {
      preparedStatement.setString(1, newGuestName);
      preparedStatement.setInt(2, newRoomNumber);
      preparedStatement.setString(3, newContactNumber);
      preparedStatement.setDate(4, Date.valueOf(newReservationDate));
      preparedStatement.setInt(5, reservationId);
      int rows= preparedStatement.executeUpdate();
      if (rows > 0) {
        System.out.println("Reservation updated successfully!");
      } else {
        System.out.println("Reservation update failed.");
      }
    } catch (SQLException e) {
      System.out.println("Error while updating reservation.");
      e.printStackTrace();
    }
  }

  private static void deleteReservation(Connection connection, Scanner scanner) {
    System.out.print("Enter reservation ID to delete: ");
    int reservationId = scanner.nextInt();
    if (!reservationExists(connection, reservationId)) {
      System.out.println("Reservation not found for the given ID.");
      return;
    }
    String sql = "DELETE FROM booking_details WHERE reservation_id = ?";
    try (PreparedStatement preparedStatement = connection.prepareStatement(sql)) {
      preparedStatement.setInt(1, reservationId);
      int affectedRows = preparedStatement.executeUpdate();
      if (affectedRows > 0) {
        System.out.println("Reservation deleted successfully!");
      } else {
        System.out.println("Reservation deletion failed.");
      }
    } catch (SQLException e) {
      e.printStackTrace();
    }
  }
  private static boolean reservationExists(Connection connection, int reservationId) {
    String sql = "SELECT reservation_id FROM booking_details WHERE reservation_id = ?";

    try (PreparedStatement preparedStatement = connection.prepareStatement(sql)) {
      preparedStatement.setInt(1, reservationId);

      try (ResultSet resultSet = preparedStatement.executeQuery()) {
        return resultSet.next();
      }
    } catch (SQLException e) {
      e.printStackTrace();
      return false;
    }
  }
  public static void exit()throws InterruptedException {
    System.out.print("Thank You For Using the Hotel Management System!!!,Successfully exited from the system 👋");
  }
}
```