SUBMITTED BY :CS-6A SUBMITTED TO: Ms.SHIVANI

1.ANUSHREE GHOSH(2100290120041)

2.AKANKSHA GUPTA(2100290120022)

3.ANANYA SRIVASTAVA(2100290120028)

**HOTEL RESERVATION SYSTEM**

****

**HOTEL RESERVATION SYSTEM**

**The main objective of creating a Hotel Reservation System Java Database Connection project is**

A hotel reservation system, or hotel reservation software, is a technology platform enabling hoteliers to accept direct bookings through the hotel’s website and various distribution channels.

Functional Features of a Hotel Reservation System:

* Associate every online booking with an account
* Limit every account to a single user
* Enable users to search and find the most relevant booking options
* Accept date and time to check available rooms for that particular time
* Booking confirmation should be sent to the specified contact details
* Calculate and display accommodation charges and other utilities
* Cancel bookings
* Display and change records of guests

Choosing the right hotel reservation system is of utmost importance. To be sure of your choice, it is always advisable to look for these essential features:

**a) Simplified Interface**:

A good reservation system will always have a simplified interface that features minimum steps required to complete the booking.

**b) Smooth Setup:**

A hotel booking system is undoubtedly a hefty investment, which is why you must opt for one that has a smooth setup process to avoid any further complications.

**c) Easy Integration with Other Systems:**

It is important to select a hotel reservation system that is compatible with other systems, including PMS, Channel Manager, Rate Management System, etc.

**d) Offers Cancellation**:

A good booking engine will always feature a cancellation policy, enabling guests to cancel their bookings without having to worry about paying a cancellation fee or losing the entire cost of the stay. It’s these little things, that go a long way in making the booking experience enhanced.

**1. CREATION OF TABLES**

**1. 1 CREATING Reservations TABLE7**

CREATE TABLE reservations(

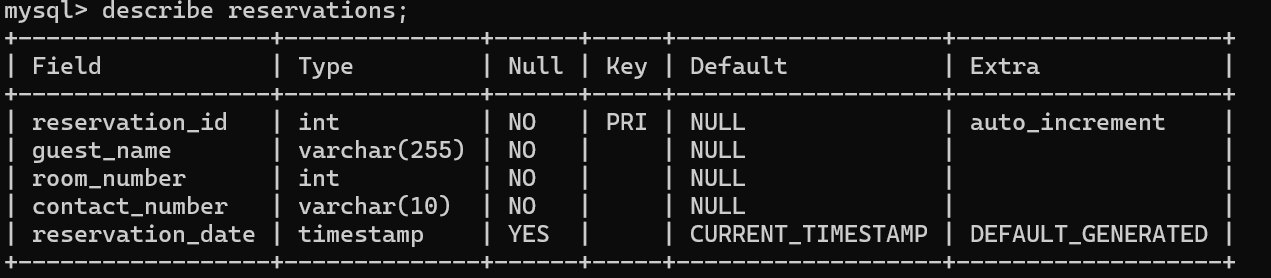
-> reservation\_id INT AUTO\_INCREMENT PRIMARY KEY,

-> guest\_name VARCHAR(255) NOT NULL,

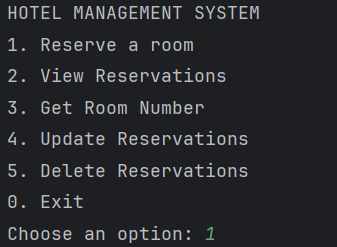
-> room\_number INT NOT NULL,

-> contact\_number VARCHAR(10) NOT NULL,

-> reservation\_data TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

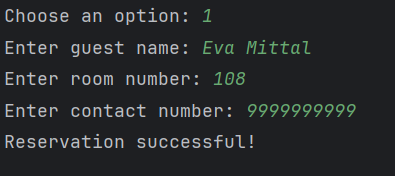
    -> );

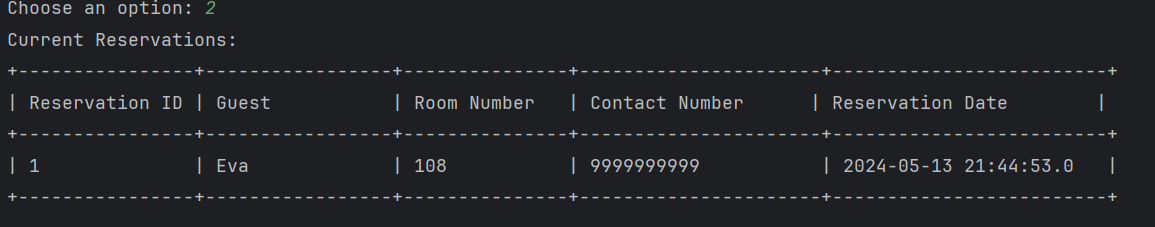
**1.2. RESERVATION INTERFACE**

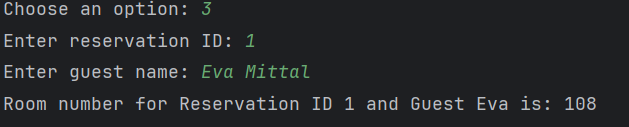
****

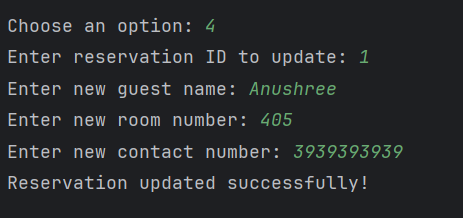
,

**1.12.1 Reserve a room**

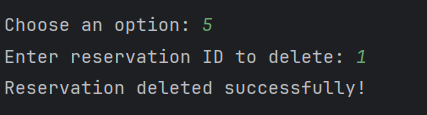
****

**1.2.2 View Reservations**

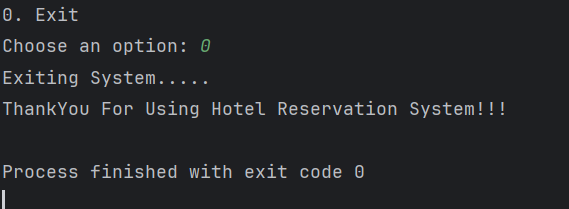
**1.2.3 Get Room Number**

**1.2.4 Update Reservations**

**1.2.5 Delete Reservations**

****

**1.2.6 Exit()**

****

**2. Java DataBase Connection(JDBC) ESTABLISHMENT**

To establish a JDBC connection between MySQL and IntelliJ IDEA java IDE using MySQL Connector 8.0, you'll need to follow these steps:

**2.1 Download MySQL Connector/J:**

Visit the MySQL website or Maven repository to download the MySQL Connector/J library (JAR file). Make sure you select the appropriate version compatible with your MySQL server version and IntelliJ IDEA.

**2.2 Add MySQL Connector/J to your IntelliJ IDEA project:**

Open your IntelliJ IDEA project.

Create a new directory named "lib" (if it doesn't already exist) within your project.

Copy the MySQL Connector/J JAR file into this "lib" directory.

**2.3 Configure IntelliJ IDEA to use MySQL Connector/J:**

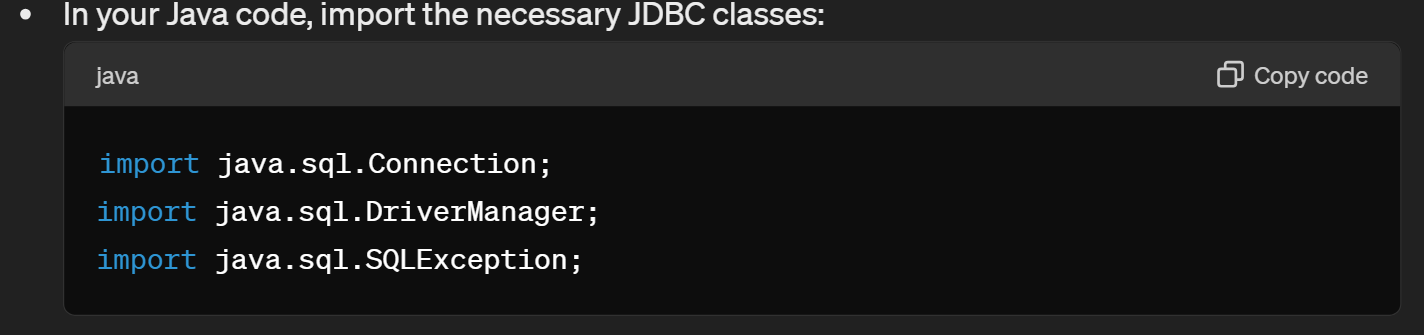
In IntelliJ IDEA, right-click on your project and select "Open Module Settings" (or press F4).

Navigate to the "Dependencies" tab.

Click the "+" icon and select "JARs or directories."

Browse to the location of the MySQL Connector/J JAR file in your project's "lib" directory and select it.

Click "OK" to add the JAR file to your project's dependencies.

**2.4 Write Java code to establish a JDBC connection**:

Write the code to establish a JDBC connection to your MySQL database:

Java

public class Main {

public static void main(String[] args) {

Connection connection = null;

try {

// Load the MySQL JDBC driver

Class.forName("com.mysql.cj.jdbc.Driver");

// Establish the connection

String url = "jdbc:mysql://localhost:3306/your\_database\_name";

String username = "your\_username";

String password = "your\_password";

connection = DriverManager.getConnection(url, username, password);

// Connection successful

System.out.println("Connected to the database!");

} catch (ClassNotFoundException e) {

e.printStackTrace();

} catch (SQLException e) {

e.printStackTrace();

} finally {

try {

if (connection != null) {

connection.close();

}

} catch (SQLException e) {

e.printStackTrace();

}

}

}

}



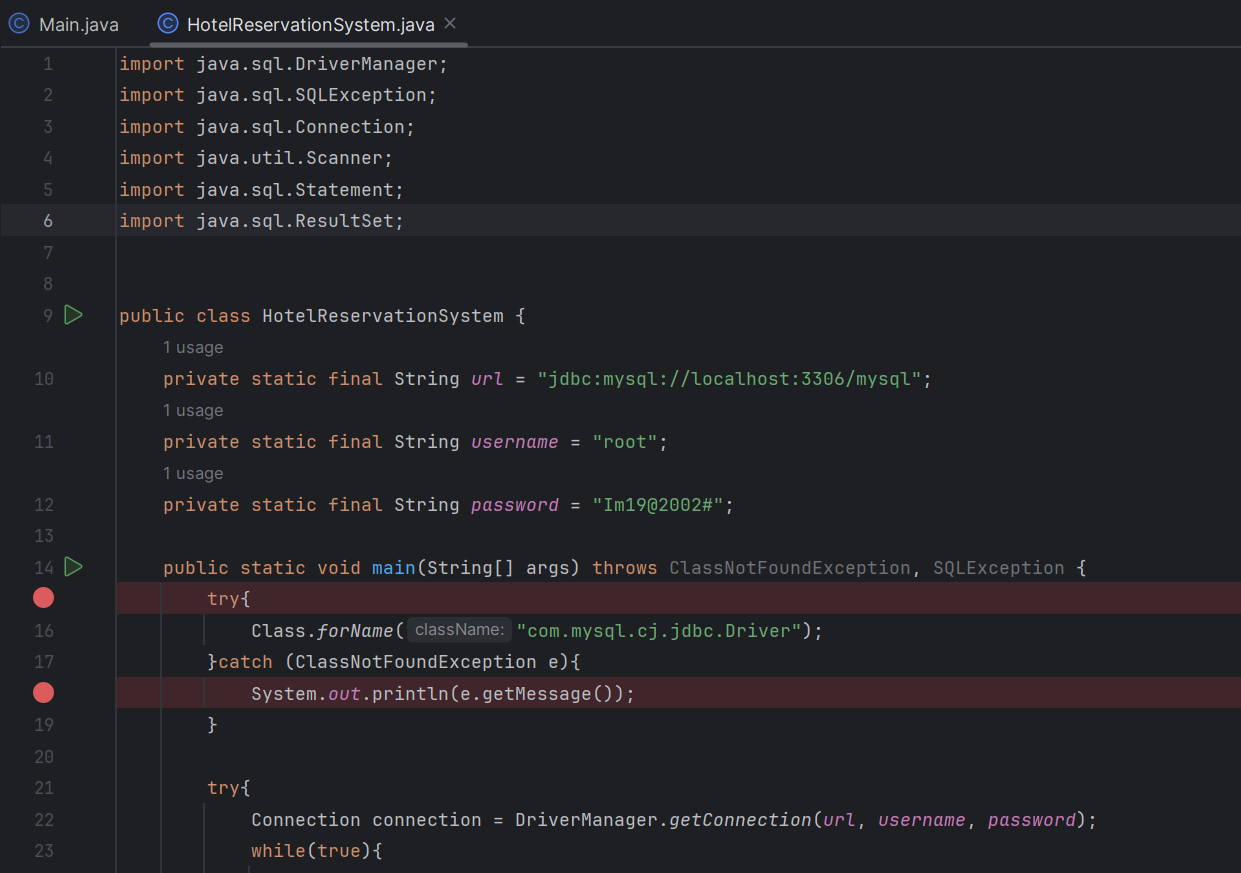
**2.5 Replace "your\_database\_name", "your\_username", and "your\_password" with your actual MySQL database name, username, and password respectively.**

**2.6 Run your Java code to test the JDBC connection:**

Execute your Java code in IntelliJ IDEA. If everything is configured correctly, you should see the message "Connected to the database!" indicating that the JDBC connection was successfully established.

By following these steps, you should be able to establish a JDBC connection between MySQL and IntelliJ IDEA using MySQL Connector 8.0.

**3. SOURCE CODE OF THE PROJECT**

****

