PROJECT SPECIFICATION

Hotel Reservation Application

Object-Oriented Programming

CRITERIA	MEETS SPECIFICATIONS
Create classes to support Polymorphism	• Create and use Java Interface classes to support polymorphism: The hotel reservation application contains the IRoom interface, which is implemented by the `Room` class. • Subclass a parent class to support polymorphism: The `FreeRoom` class extends the `Room` class.
Use of Access Modifiers	• Define class variables as private and provide accessor and mutator methods to get and manipulate class variables: There is at least one example of the model classes (`Room` , `Customer` , `Reservation`) using data encapsulation. • Use variable access modifiers (such as public, private and final) to modify access: The application contains at least one example of using each of the following access modifiers: 'public', 'private', and 'final'. Note: You should use the `final` for the data model classes variables and public methods.
Override an object `toString` method to provide a better description of the object.	There is at least one example of the model classes (`Room` , `Customer` , `Reservation`) overriding the `toString` method.

Processing and Storing Data

CRITERIA	MEETS SPECIFICATIONS
Store and process data for an application in a Collection.	Collections are used to store data for: • Room • Customer • Reservation The collection type chosen for rooms ensures that two rooms cannot be booked at the same time. Tip: A user should not be able to book a single room twice for the same date range.
Use the `static` keyword to create singleton objects.	All of the service classes use `static` references to create singleton objects. Tip: You need to create a static reference for the service class.

Core Java Concepts

CRITERIA	MEETS SPECIFICATIONS
Use Regular Expressions to validate String input.	The `Customer` class should contain at least one example of validating a String to ensure that it has valid email address syntax.
Use of Java Types and Enumeration	• Use Java types to store and process data: The application uses different Java types (String, Double, and Dates) to store data

Use of Java Types and Enumeration Classes	• Use Java types to store and process data: The application uses different Java types (String, Double, and Dates) to store data on objects. • Use Date and Calendar objects to store and process dates: The `Reservation` class uses Date objects for check-in and check-out dates. • Create Enumeration classes: The application contains the enumeration class `RoomType`.
Use `Exceptions`, `try` and `catch` blocks to handle error flow.	The application contains at least one example of using `Exceptions` to validate input and `try` and `catch` blocks to handle error flow without crashing the application.
Use a `switch` statement to process user input.	The application UI uses a `switch` statement to handle the user input flow.

Suggestions to Make Your Project Stand Out!

1. Customize the find-a-room method to search for paid rooms or free rooms.
2. Provide a menu option from the Admin menu to populate the system with test data (Customers, Rooms and Reservations).
3. Allow the users to input how many days out the room recommendation should search if there are no available rooms.