UDACITY

‹ Return to Classroom

DISCUSS ON STUDENT HUB

Hotel Reservation Application

REVIEW

CODE REVIEW 4

HISTORY

Meets Specifications

Congratulations !!

You have done a great job.Making the required changes and submitting the project in such a short span of time clearly shows your hardwork and dedication towards the project.The project meets all the required points. I like the fact that **you have split your code into methods and functions makes the code more clean and easily readable** 👏

You did an excellent job on this project, I have enjoyed running and interacting with it. And you were able to apply all the changes quickly and efficiently. 👍👍

I hope this project was fun. Take a small break and prepare for the next battle.
We look forward to receiving your future project submissions soon .
Keep it Up !! Wishing you Best of Luck for journey ahead :)

Resources :

- Here is a good read about Future of Java.
- Some useful Tips and Tricks for Beginners

PS: If you have any doubts regarding any of the concept, feel free to search or post a question on Knowledge where many of the fellow students and mentors may have faced the same situation before and would have provided the appropriate steps to resolve it.

Have a Good Day and Stay Safe ✌

Keep Learning and Stay Udacious

Object-Oriented Programming

The hotel reservation application contains the IRoom interface , which is implemented by the `Room` class.

This specification was already passed in the previous submission

The `FreeRoom` class extends the `Room` class.

This specification was already passed in the previous submission

There is at least one example of the model classes (`Room` , `Customer` , Reservation`) using data encapsulation.

This specification was already passed in the previous submission

There is at least one example of the model classes (`Room` , `Customer` , `Reservation`) overriding the `toString` method.

This specification was already passed in the previous submission

There is at least one example of the model classes (`Room` , `Customer` , `Reservation`) overriding the `equals` and `hashcode` methods.

`equals` and `hashcode` methods have been overriden for all the model classes (`Customer` , `Reservatiom` and `Room`) class ✅

The application contains at least one example of using each of the following access modifiers: 'public', 'private' and 'final'.

This specification was already passed in the previous submission

Processing and Storing Data

Collections are used to store data for:

- Room
- Customer
- Reservation

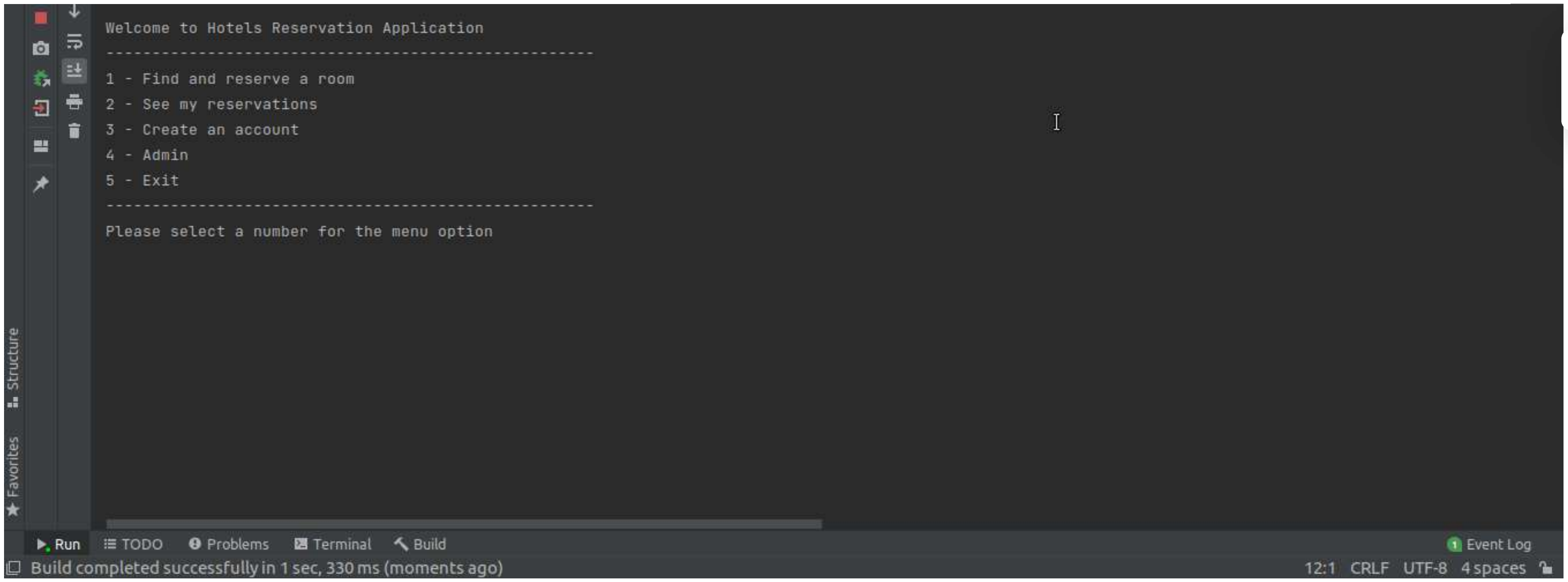
The collection type chosen for rooms ensures that two rooms cannot be booked at the same time.

Collection has been used ✔

HashMap, HashSet and ArrayList have been used to store the Customer,Reservation and Room data. ✅

Duplicate Reservations not allowed ✔

The application correctly handles **duplicate room scenario**. Same room can be booked only once for a given date range, as can be seen below ✅



```
Welcome to Hotels Reservation Application
----------------------------------------------------
1 - Find and reserve a room
2 - See my reservations
3 - Create an account
4 - Admin
5 - Exit
----------------------------------------------------
Please select a number for the menu option
```

The `ReservationService` contains `for` or `while` loops that are used to iterate over and process data in order to do the following:

- Search for available rooms
- Search for recommended rooms

This specification was already passed in the previous submission

All of the service classes use `static` references to create singleton objects.

This specification was already passed in the previous submission

The `ReservationService` contains at least one example of using each of the following method access modifiers:

- `public`
- `private`
- `default`

This specification was already passed in the previous submission

Core Java Concepts

The `Customer` class should contain at least one example of validating a String to ensure that it has valid email address syntax.

This specification was already passed in the previous submission

The application contains the enumeration class `RoomType`.

This specification was already passed in the previous submission

The Reservation class uses Date objects for check-in date and check-out date.

This specification was already passed in the previous submission

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.

This specification was already passed in the previous submission

The application uses different Java types (String, Double and Dates) to store data on objects.

This specification was already passed in the previous submission

The application UI uses a `switch` statement to handle the user input flow.

This specification was already passed in the previous submission

DOWNLOAD PROJECT

4 CODE REVIEW COMMENTS ›

RETURN TO PATH