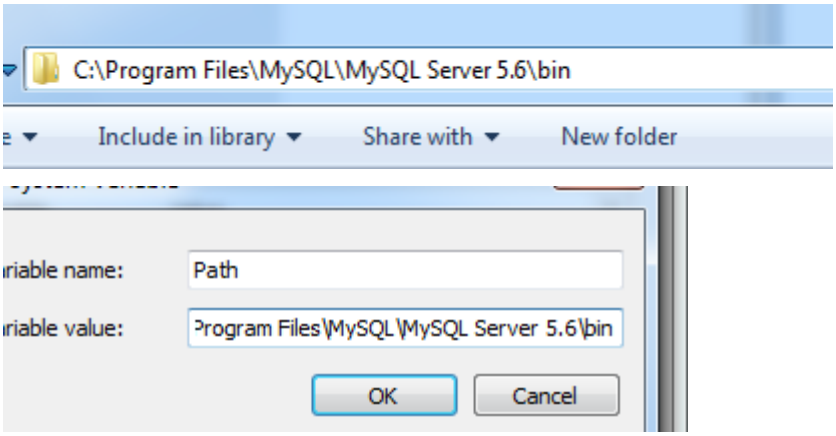
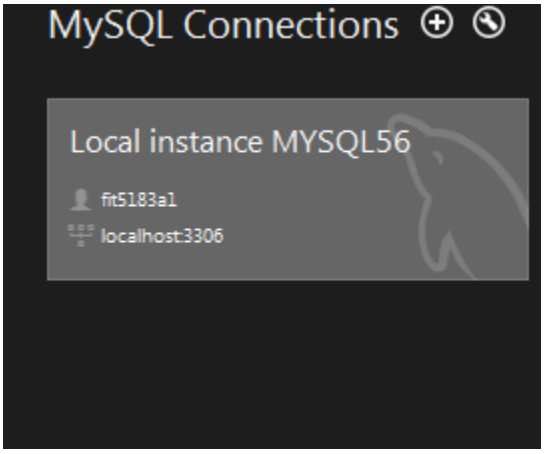
Design Document: Hotel Booking Broker System

Deployment instruction:

The most import thing is to import the database before you execute this system. First of all, configure the environment of the mySQl. find your mySQL path add to the environment Variables



Then create a no password account "**fit5183a1**" on your MySQL workbench.

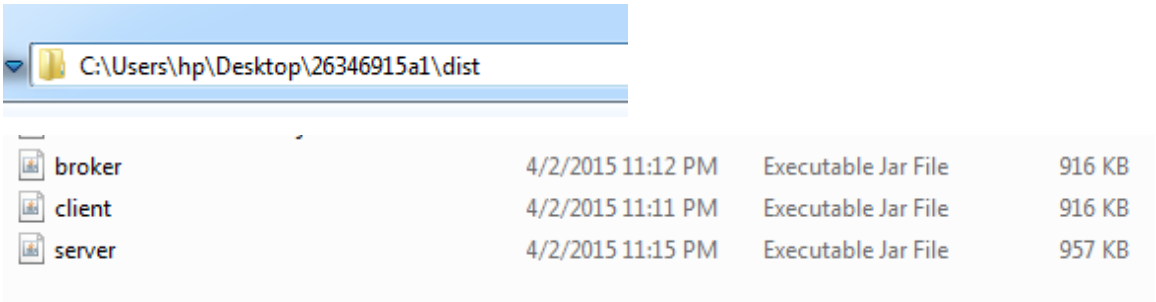


enter the below command you can successfully import the id26346915.sql file.

User manual or usage guidelines:

This is the guidelines under the DOS environment ,please pay attention to every steps before you execute the system.

Before you execute this system make sure that you have 3 jar package in your specify directory.

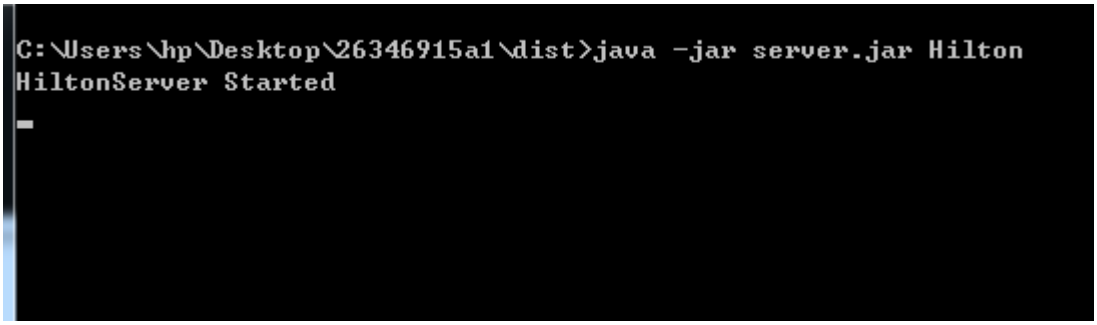


1.First of all, you should start 3 hotels server(your must specify which hotel your want to start, so at the end of the ".jar" specify the hotel argument)

start Jinjiang hotel server:

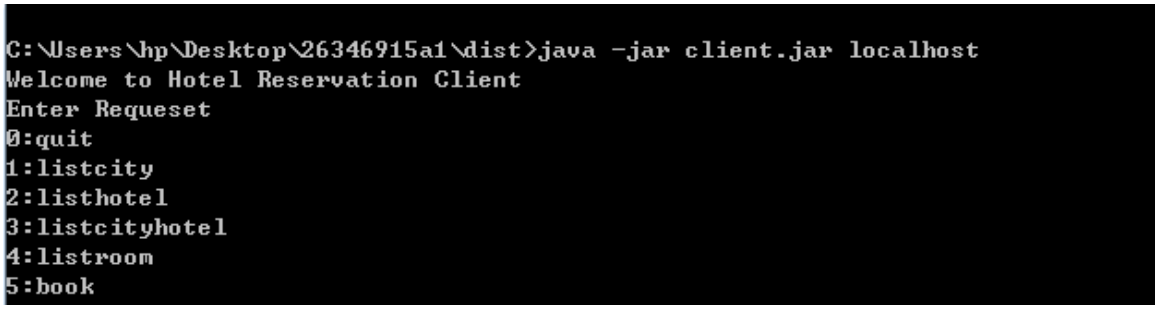
start Regent hotel server:

start Hilton hotel server:



2. start the Broker server:

3. execute the client(just start one client to show how this system work ,you can start more than one client)

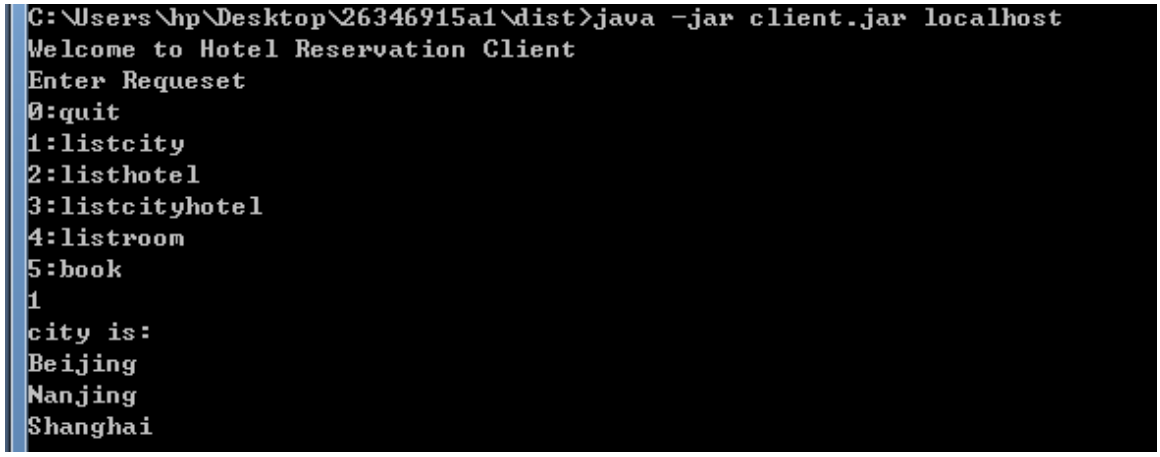


from the figure you can see that client should specify the argument too, because the client is executed in my own computer, so the argument is "localhost".

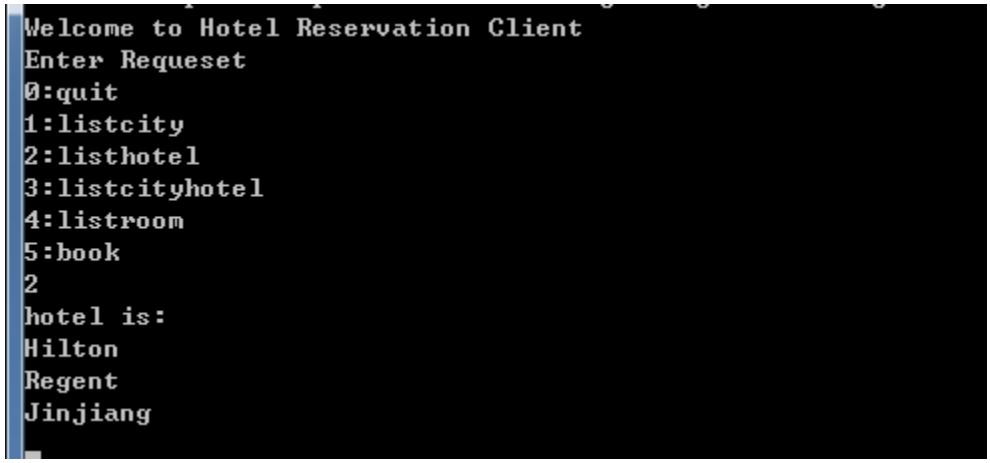
4. Function demonstrate

We can know that there are five function keys from the third step, Here we demonstrate one by one.

list city:



list hotel:



list city hotel:

```
Enter Requeset
0:quit
1:listcity
2:listhotel
3:listcityhotel
4:listroom
5:book
3
please enter the city
Shanghai
hotel is:
Jinjiang
```

listroom:


```
4
please enter the city
Shanghai
please enter the hotel
Jinjiang
Shanghai        Jinjiang        140     3 star  single
Shanghai        Jinjiang        200     3 star  double
```

book:

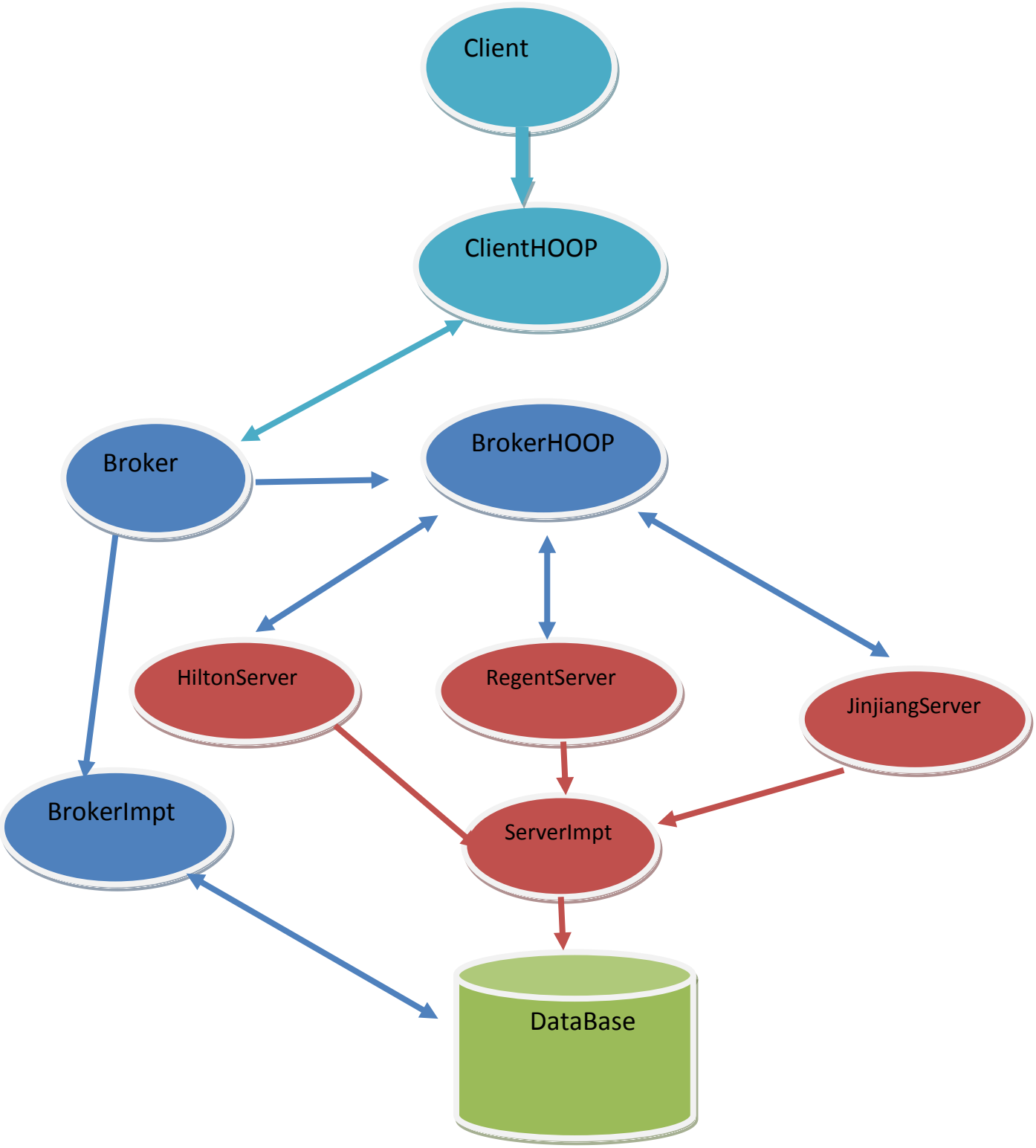
```
5
please enter the city
Beijing
Please enter the hotel name
Hilton
please enter the room type you want book<single or double>
single
please enter the check in date (formate:yyyy-mm-dd)
2015-04-01
please enter the check out date (formate: yyyy-mm-dd)
2015-04-05
the room is availabe and your ID is315032
please enter your information
please enter your name
Zhushuai
please enter your credit number
342222199202194430
please enter your email
1052785913@qq.com
booking success!
```

because this system does not have the function of check booking ,so from the database we can see the state of booking.

id	check_in	check_out	hotel	city	style
315032	2015-04-01	2015-04-05	Hilton	Beijing	single
(NULL)	*(NULL)*	*(NULL)*	*(NULL)*	*(NULL)*	*(NULL)*

city	hotel_name	price	rate	style	state
Beijing	Hilton	150	3 star	single	9
Beijing	Hilton	200	3 star	double	10
Beijing	Regent	170	4 star	single	10
Beijing	Regent	240	4 star	double	10

System architecture diagram:



From the diagram we can see this Hotel Booking Broker System is **a typical 3-tier system** ,Client- Broker-HotelServer 3 tier mechanism .

Client as the user Interface to implement the command your want to send , it does not communicate directly with the Broker but by the **ClientHOOP. ClientHOOP** will build a path(Socket) to send or receive message from the **Broker**.

Broker as the server to communicate directly with the client and it also plays the role of the client . When user want to list city or hotel ,**ClientHOOP** send the request to **Broker**, it no need to gain the information of the Database from the hotel server. The easy way is that **Broker** as the server to accept the request message and gain the information from the database, in order to well implement the independence of the code , in the **Broker** we extract the function of accessing the database as the **BrokerImpt**. But if the user want to book room or list room ,the Broker must be a client to send request to the 3 hotel servers. Extract the function of communication with the hotel server as the **BrokerHOOP** which similar to the **ClientHOOP**. To get the information from the database the 3hotel server extract this function as the **ServerImpt**.

Table on message types:

request	Send(client)	Send(Broker)	Receive(Broker)	Receive(server)	Resp(broker	Res(server)
List city	"1"\r\n		"1"\r\n		"City info"	
List hotel	"2"\r\n		"2"\r\n		"Hotel info"	
List city hotel	"3"\r\n		"3"\r\n		"Hotel info"	
listroom	"4"\r\n	"listroom"	"4\r\n"	"listroom"	"roominfo"	"roominfo"
Book(room not availablle)	"5"\r\n	"bookroom"\r\n	"5"\r\n	"bookroom"\r\n	**-1**	**-1**
Book (avaialbe) **Enter user info**	"5"\r\n "user id,email cno..."	"bookroom"\r\n insertInfo	"5"\r\n "user id,email cno..."	"bookroom" Insert info	Userid True if success	Userid True or flase
Quit	**"0"**	**"quit"**				

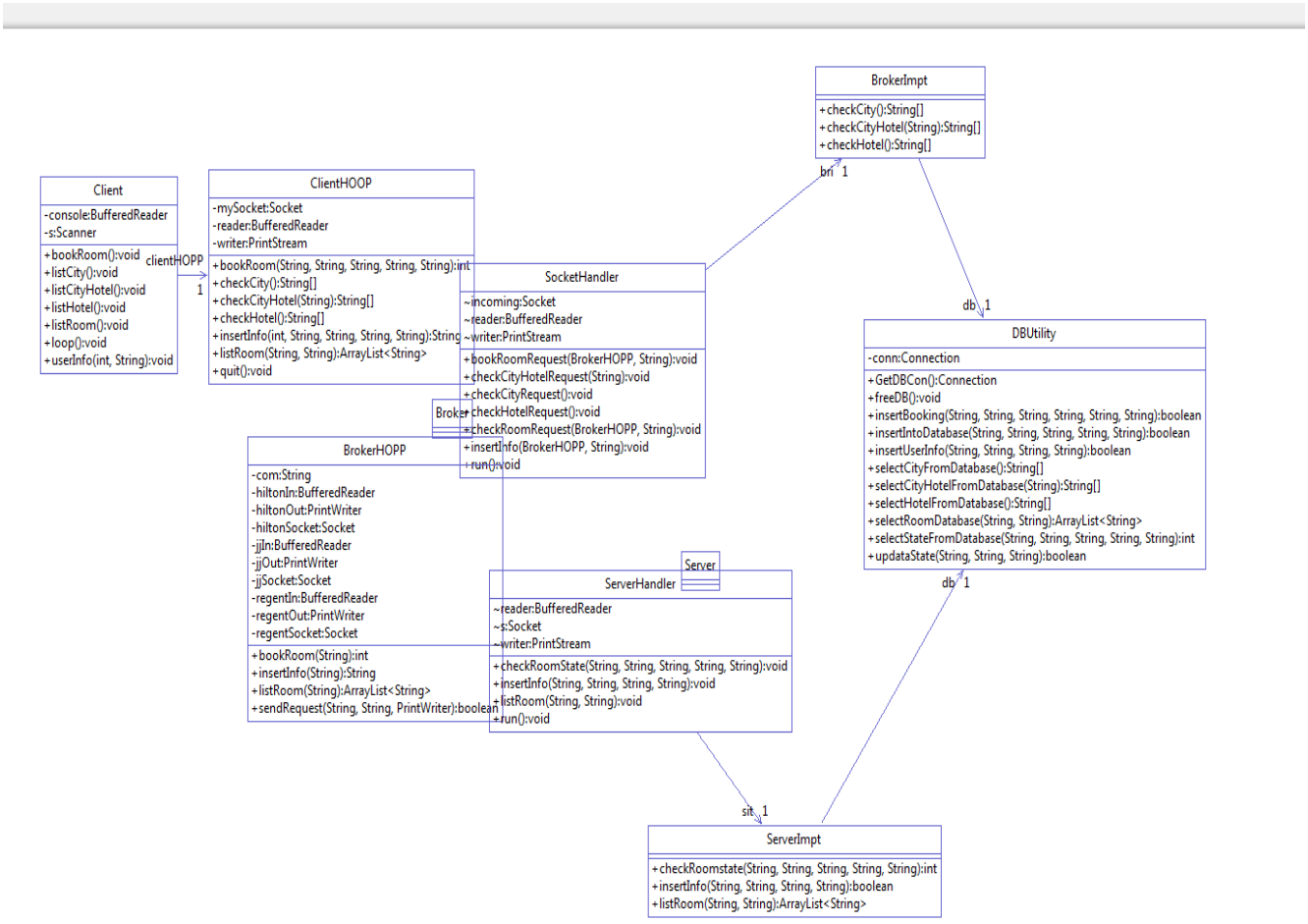
a.list city , hotel and cityhotel ,send the request to the Broker , in order to simplify the operate , use number 1,2,3 instead of entering the command , so client send "1","2","3",and Broker receive "1","2","3". When Broker get the command and get the information from the database to respond.

b. list room : client send "4" to indicate the function of list room, Broker could not directly access the database ,so Broker send the "list room" message to hotel server, then server access the database and respond to the client.

c. book:(not available) when send the request about the book room, the broker will send this request to the server, then server will check the availability of the room and send the boolean type value -1 to respond the client.

d. book(available) After the server check the room state and know there are some room available. Then server will randomly generated six figures as the user id respond to the client, once the client once get this id then enter the user information as the request send to server and insert into database.

UML class diagram of key components:



code explanation:(use two example two illustrate the interaction of the classes)

1. when **client** want to list city, the client class will call the function of the listcity() then new a **ClientHOOP** instance to call the function of checkcity() in class **ClientHOOP**. Then ClientHOOP use the inputStream and OutputStream send request to the **Broker**, Broker use the class SocketHander to accept the ClientHOOP request , and in order to responds to ClientHOOP , it news a **BrokerImpt** instance to call the method of CheckCity(), the **BrokerImpt** new a **DBUtility** instance to get the information from the database.

2. when client want to book room same step to send request to Broker, the Broker could not respond directly to client but plays the role of client to send request to the hotel server

through new a **BrokerHOOP** instance . Then the BrokerHOOP instance to call its checkRoomState() method to check whether the room which the user want to book is available or not. To implement the method of check room it must access the database via new **serverImpt** instance to call the method of checkRoomstate(). If the room is available then the client will receive the user ID and enter the user information to send the database through same rules of communication.