House Management System

Project Objective:

Create a console based Java application that would allow a user of a House Management System to perform the following operations

- Create a new Rental Property into the system [Posting availability of a new property that is available for rent]
- The user will receive the following details from the customer
  - The rental amount
  - The number of bedrooms
  - City name
  - Location name
  - The property id needs to be computed

Project Design:

A. Database Design:

1. **Create a new user in database [To be done in the backend by using sql commands]**
   a) **Note:** Do **NOT** use the default scott/tiger account of oracle for this project. You will have to create a new user in the below mentioned format.
   b) Username/password : B<batchnumber><employeeid>
   For example, if your batch number is **39806** and Employee number is **12345**, then the oracle user should be **B3980612345** and the password should be **B3980612345**
   c) For JDBC connection, only use **XE** as service name and **1521** as port number

2. **Steps for creating a new user**
   a) Open command prompt
   b) Sqlplus / as sysdba
   c) Create user <username> identified by <password>; [For example to create a user named "test" with password "test" : create user test identified by test;]
   d) Grant connect,resource to <username>; [E.g: grant connect,resource to test;]
   e) Commit;
   f) Exit;

3. **Create Table [To be done using sql commands, after logging-in as the new user that has been created in above step]**

Table Name: RENTAL_TBL

Column	Datatype	Description
PROPERTYID	VARCHAR2(20)	PRIMARY KEY (auto generated)
RENTALAMOUNT	NUMBER(10, 2)	NOT NULL
NOOFBEDROOMS	NUMBER(2)	NOT NULL
LOCATION	VARCHAR2(20)	NOT NULL
CITY	VARCHAR2(20)	NOT NULL

Sequence Name: RENTAL_SEQ

Sequence Name	Minimum Value	Max Values	Incremental value	Start Value
Rental_seq	1000	9999	1	1000

B. System Design:

Name of the package	Usage
com.anna.hms.service	This package will contain the class that displays the console menu and takes the user input.
com.anna.hms.bean	This package will contain the bean class
com.anna.hms.dao	This package will contain the class that will do the database related operations
com.anna.hms.util	This package will contain a class to establish database connection and also another class that handles the user defined exception.

Package: com.anna.hms.util

Class	Method and Variables	Description
DBUtil		DB connection class
	public static Connection **getDBConnection**()	Establish a connection to the database and return the java.sql.Connection reference
InvalidCityException		User defined Exception class
	public String toString()	Returns "**INVALID CITY**"

Package: com.anna.hms.bean

Class	Method and Variables	Description
RentalPropertyBean		**Class**
	private float rentalAmount;	
	private int noOfBedRooms;	
	private String location;	
	private String city;	
	private String propertyId;	

Package: com.anna.hms.dao

Class	Method and Variables	Description
RentalPropertyDAO		DAO class
	public String generatePropertyID(String city)	This method should return the propertyId which is auto generated using sequence. **Format:** First 3 letters of the city in uppercase followed by the 4 digit auto generated number. E.g. CHE1000 – if the city name is Chennai
	public int createRentalProperty(RentalPropertyBean bean)	• This method should invoke the **generatePropertyID** method of this class, receive the id and initialize the bean with the id value. • It should then insert the bean's properties into the rental_tbl and should return the count of the records inserted • If there is some Exception(**could be SQLException or any other Exception**), the function should **return -1**.

Package: com.anna.hms.service

Class	Method and Variables	Description
RentalPropertyService		Main class
	public static void **main**(String[] args)	The code that is needed to test your program goes here. A sample code is shown at the end of the document.
	public String addRentalProperty(RentalPropertyBean bean)	• It returns "**NULL VALUES IN INPUT**" if city or location is null • It returns "**INVALID INPUT**" under the following conditions ➢ Length of city or location is 0 ➢ No of bedrooms/Rental amount is 0 • This method will call validateCity method for validating city. If city is not either Chennai or Bengaluru (not case sensitive) the method should return "**INVALID CITY**" In all other conditions this method should call createRentalProperty method of DAO and returns "**SUCCESS**" if record is inserted else returns "**FAILURE**" if the record cannot be inserted
	public void validateCity(String city) throws InvalidCityException	• This method will throw InvalidCityException if the provided city name is not either chennai or bengaluru (case insensitive)

Main Method:

You can write code in the main method and test all the above test cases. A sample code of the main method to test a test case is shown below for your reference.

```
public static void main(String[] args) {

RentalPropertyBean bean=new RentalPropertyBean();
bean.setCity("Chennai");
bean.setLocation("Velachery");
bean.setNoOfBedRooms(2);
bean.setRentalAmount(15000);
RentalPropertyService service=new RentalPropertyService();
System.out.println(service.addRentalProperty(bean));
}
```