**Important Instructions:**

* Please read the document thoroughly before you code.
* Import the given skeleton code into your Eclipse.
* Do not change the Skeleton code or the package structure, method names, variable names, return types, exception clauses, access specifiers etc.
* You can create any number of private methods inside the given class.
* You can test your code by running main() method of the program
* Using **Spring Maven** to develop the application

**General Design Constraints:**

* Ensure that all the Java Coding Standards are followed.
* Assume that the method inputs are valid always, hence exceptional blocks are not needed to be included in the development.

**Problem Statement**:

A TakeHome company requires billing software for its super market where different types of products are sold to the customers on retail basis.

This billing software should help

1. The cashier to see the product information by entering the product code
2. Calculate the total cost for each product based on the quantity
3. Display the final bill to user
4. Throw exception for invalid details

**Skeleton File for Development:**

Import the below attached skeleton code into your eclipse project and implement the required functionalities

**Technical Requirements:**

**Requirement 1:Database design**

Use createScripts.sql file for the database requirement



**Requirement 2:Bean class**

Create a Product bean class(com.cts.bean) with following members.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Variable Name** | | | | **Data type** | | | |
| product\_code | | | | Int | | | |
| product\_name | | | | String | | | |
| product\_category | | | | String | | | |
| product\_description | | | | String | | | |
| product\_price | | | | double | | | |
| **Requirement 3:DAOInterface**  Write a ProductDao interface (com.cts.dao) with following members. | | | | | | | | | | | | |
| **Interface Name** | | **Method Name** | | | | | **Input Parameters** | | | | **Output Parameters** | |
| ProductDao | | getProductDetails(); | | | | | Int product\_code | | | | Product | |
|  | | | | | | | | | | | | |
| **Requirement 4:DAO class implantation**  Write a ProductDaoImpl class (com.cts.dao) which implements ProductDao interface with following members. | | | | | | | | | | | | |
| **Class Name** | **Method Name** | | **Input Parameters** | | | **Output Parameters** | | | | **Logic** | | |
| ProductDaoImpl | getProductDetails | | Int product\_code | | | Product | | | | Query to DB select a product by product\_code using jdbcTemplte.queryForObject() method and return the object. | | |
| **Requirement 5:Service class Impl**  Write a ProductService class (com.cts.service) which has following members. | | | | | | | | | | | | | |
| **Class Name** | | **Method Name** | | | | **Input Parameters** | | | | **Output Parameters** | | | **Logic** | |
| ProductService | | getProductDetails | | | | Int product\_code | | | | Product | | | Invoke the ProductDao getProductDetails() return the object. | |
| ProductService | | calculatePrice() | | | | Int quantity,double product\_price | | | | double | | | Use quantity and price calculate the total amount and return the value. | |
| ProductService | | validateProductCode() | | | | Int product\_code | | | | boolean | | | Check product\_code should be >0 and 4 digit number return true else return false | |
| ProductService | | validateQuantity() | | | | int quantity | | | | boolean | | | Check quantity should be >0 return true else return false | |

**Requirement 6:Exception Class**

Write a BillingException class (com.cts.service) which extends Exception classraise the exception as per the requirement.

**Requirement 7:MainClass**

In main method class to test your application. Use switch case to perform as per requirement.

**Note:** Throw exception called **BillingException** with appropriate error message for the following cases and get input until user enters the valid information

* Quantity should not be lesser than or equal to zero otherwise raise the exception
* Product code should be 4 digit positive number otherwise raise the exception

**SAMPLE INPUT and OUTPUT**

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

1. Generate Bill by entering code and quantity

2. Exit

1

Enter product\_code

Invalid product\_code, it should be >0 and 4 digit +ve number

896

Invalid product\_code, it should be >0 and 4 digit +ve number

Enter product\_code

-75

Invalid product\_code, it should be >0 and 4 digit +ve number

Enter product\_code

0

Invalid product\_code, it should be >0 and 4 digit +ve number

Enter product\_code

1002

Enter quantity

-9

Enter quantity

Invalid quantity, it should be >0

0

Invalid quantity, it should be >0

Enter quantity

2

Product Name: LEDTV

Product Category: electronics

Product Description: TV

Product Price(Rs): 45000.0

Quantity: 2

Total Bill Amount: Rs.90000.0

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

1. Generate Bill by entering code and quantity

2. Exit

3

Enter option 1 to 2 only

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*