DEV BHOOMI — UTTARAKHAND — UNIVERSITY

Table of Contents

BACKGROUND 1

1. BUSINESS REQUIREMENTS 1
2. THE CONSOLE APPLICATION 2
3. MENU SYSTEM 2
   - 3.1. Class Design for Customer Registration Module 2
   - 3.2. Class Design for Courier Order Booking Module 3
   - 3.3. Class Design for Courier Dispatch Responsibility Generation Module 4
   - 3.4. Class Design for Courier Delivery Module 4
   - 3.5. Class Design for Invoice Generation Module 5
   - 3.6. Class Design for SterlingDAO class 6
4. GUIDELINES 6

Background

This document contains the high level design of the project that has to be executed in order to complete the course Java Programming.

The project is to automate Sterling Courier, to provide better service to the customers and edge over their competitors.

Business Requirements

The Business Requirements of the project is as follows. Sterling wants you to implement the following modules.

- Customer Registration
- Courier Order Booking
- Courier Delivery List Generation
- Courier Delivery
- Invoice Generation

Every day, Courier orders are accepted from the registered customers. The Manager or Clark would assign the orders to the delivery boys for dispatching. The courier is allocated to a delivery boy, if the delivery boy belongs to the region where the courier order has to be dispatched. Once assigned, the delivery boy would dispatch the courier and updates the status in the application. The delivery boy keeps dispatching the courier till the recipient accepts the courier or has been rejected due to reasons like the recipient not accepting the courier, the recipient permanently not available in the address.

Sl	Requirement	Status
1	Customer Registration	To be implemented
2	Courier Order Booking	To be implemented
3	Courier Delivery List Generation	To be implemented
4	Courier Delivery	To be implemented
5	Invoice Generation	To be implemented
6	Console Interface to read the data for all the modules i.e. Presentation Tier	To be implemented

The Console Application

The system has to be web based application, which is part of Introduction to JEE module. The business tier classes generated as part of this project would be reused in the forthcoming projects. The console application, written as part of Java Programming module is used for the purpose of testing the classes created in Advanced Java project. As this application is for testing, client side validations are not required for any of the screens.

Menu System

This module is to provide a menu system for easily accessing the functionalities.

```
=====================================
        Sterling Courier Company
=====================================
    1. Customer Registration
    2. Courier Order Booking
    3. Courier Dispatch Responsibility Generation
    4. Courier Delivery
    5. Invoice Generation
    6. Exit

   Enter your choice :
```

Hint: To accept the choice from the console use ReadData.acceptString() method of the ReadData class, provided to you. After taking the input (which is a String) convert it to int.

Class Design for Customer Registration Module

This module is used to register the customer. This module should collect the customer's information from the user and store it in a Customer bean object.
The screen design should be as shown below:

```
        Customer Registration
-------------------------------------

Customer Name                       :David
Date Of Registration(DD-MMM-YYYY)   :10-Jul-2022
Address                             :#350, Hebbal Electronics City
City                                :Mysore
Pin                                 :570 018
Telephone Number                    :2404200
E-mail Id                           :david@yahoo.co.in
```

To read the data from the console create CustomerTester.java, which contains readCustomerInfo() method, that reads the data from the user and stores the data in the Customer bean instance.

Note: readCustomerInfo() of CustomerTester accepts the customer name, date of registration, address, city, pin, telephone number, e-mail-id from the user and then this data should be stored in the Customer bean instance.
After setting the properties of the customer bean instance, readCustomerInfo() method should pass this customer bean instance to saveCustomer() method of the SterlingDAO class(Refer to the design of SterlingDAO).

Class Design for Courier Order Booking Module

This module is used when the order has to be placed for sending the courier. The module should collect the data from the user and store it in an order bean object.

The screen design should be as shown below:

```
        Courier Order Booking
-------------------------------------

Customer Id                         :2002
Date Of Order(DD-MMM-YYYY)          :10-Jul-2022
Recipient Name                      :Michelle
Recipient Address                   :#F10,St.Mark's Road
Recipient City (D, P, B, C)         :D
Courier Weight                      :4.25
Description                         :Envelope
```

To read the data from the console create OrderTester.java, which contains readOrderInfo() method, that reads the data from the user and stores the data in the Order bean instance.

Note: readOrderInfo() of OrderTester accepts the customerId, orderDate, recipientName, recipientAddress, recipientCity (D-Delhi / P-Pune / B-Bombay / C-Chennai), WeightOfTheParcel, Courier Description from the user and then this data should be stored in the Order bean instance.

After setting the properties of the order bean instance, readOrderInfo() of OrderTester class, should pass this order bean instance to saveOrder() method of the SterlingDAO class(Refer to the design of SterlingDAO).

Class Design for Courier Dispatch Responsibility Generation Module

This module is used to generate the courier courier delivery list for a particular day. The assumption in this project is that there is only one delivery boy per region. Every day, the manager invokes the module to assign the order's that are to be dispatched (Order with status 'Accepted') to the Delivery boy. Distribution list generation will be partially implemented now and the rest will be implemented as a part of advanced java project. In java project, just create the classes to take the input from the user.

The screen design should be as shown below:

```
            Courier Dispatch Responsibility Generation
      ------------------------------------------------------

    Date Of Distribution (DD-MMM-YYYY) :10-Jul-2022
```

To read the data from the console create DistributionTester.java, which contains readDistributionListInfo() that accepts the date on which the manager wants to generate the courier distribution list.

Note: readDistributionListInfo() of DistributionTester accepts the date from the user and this date should be passed to saveDistribution() of SterlingDAO class(Refer to the design of SterlingDAO).

Class Design for Courier Delivery Module

This module is used by the delivery boy to record the status of delivery of a particular order. The status of the courier might be 'Pending'(P) (if the recipient is not available currently), 'Rejected'(R) (if the recipient does not accept the courier or if the recipient is not available in the address permanently) or 'Delivered'(D). The module should store the data in the delivery bean object.

The screen design should be as shown below:

```
          Courier Delivery
    ------------------------------------

    Employee Id                     :1001
          Assigned Courier's : 2001, 2002, 2003
    Order Id                        :2003
    Date Of Delivery(DD-MMM-YYYY)   :11-Jul-2022
    Status (P, R, D)                :R
    Remarks                         :Recipient Not Available
```

To read the data from the console create DeliveryTester.java, which contains readDeliveryInfo() method, that reads the data from the user and stores the data in the Delivery bean instance.

Note: readDeliveryInfo() of DeliveryTester should read the EmployeeID(Consider that 1001 is the valid employee id and input it). The method should display the list of OrderID's that are to be delivered by the given employee (It should always display 2001, 2002, 2003. In the advanced java module this code will be modified to show the actual values). Then accept OrderId, DeliveryDate, description and status of the delivery (either 'pending', 'rejected' or 'delivered') from the user and then this data should be stored in the Delivery bean instance.

After setting the properties of the delivery bean instance, readDeliveryInfo() of DeliveryTester class, should pass this delivery bean to saveDelivery() method of the SterlingDAO class(Refer to the design of SterlingDAO).

Class Design for Invoice Generation Module

This module is used to generate the invoice for a given month and year for a particular customer. The business logic to calculate the invoice amount is as follows. The invoice amount is the total amount to be paid for each order. The invoice amount is calculated based on the weight of each courier booked by the customer in a particular month and year. Actual amount calculation will be done as a part of next modules. In the java module we need to create classes for taking input from the user.

The screen design should be as shown below:

```
          Invoice Generation
-------------------------------------

Customer Id                    :1001
Invoice Month                  :7
Invoice Year                   :2022
Description                    :Invoice for the month of July
```

To read the data from the console create InvoiceTester.java, which contains readInvoiceInfo() method, that reads the data from the user and stores the data in the Invoice bean instance.

Note: readInvoiceInfo() of InvoiceTester accepts the customer id, invoice month, invoice year and description from the user then this data should be stored in the Invoice bean instance.

After setting the properties of the invoice bean instance, readInvoiceInfo() of InvoiceTester class, should pass this invoice bean to saveInvoice() method of the SterlingDAO class(Refer to the design of SterlingDAO).

Class Design for SterlingDAO class

Note:

- Each method of this class should display a success message. E.g. saveOrder() should display "Order is saved."
- All the methods, which are returning int, should return 0, as part of java project.
- The actual functionality of each of these methods will be implemented during Advanced Java Project.

Guidelines

- All the java coding standards must be followed.
- All the modules should display appropriate messages.
- Java documentation should be generated describing all the classes and its members
- The group should finally integrate the project before evaluation
- Submission Guideline
    - Create a folder as Group<GroupNo>
        - E.g. if you belong to Group 11, the folder name should be Group11
    - Copy paste the entire project into the folder