MINI PROJECT :3

Date:25.07.2025
Name:Nancy M

Holiday Package

The tour package details like package Id, source place, destination place, basic fare and number of days for each package should be stored in a table. Calculate the package cost based on the basic fare and the number of days for each package.

Instance_variable	Data type
package_id	String
source_place	String
destination_place	String
no_of_days	int
package_cost	double

Create a model class to represent this with attributes, getters and setters.

Calculate the package cost for each package, based on the conditions given below,

noOfDays	discount %
<=5	0% (No discount)
>5 and <=8	3%
>8 and <=10	5%
>10	7%

Package Cost = ((Basic fare x number of days)-discount) + GST

The package cost should be calculated based on the basic fare and the number of days. The discount should be calculated depending on the number of days as given in the above table and deducted from the calculated package cost. Finally, a GST of 12% of the calculated package cost got after the discount, should be added to get the final package cost.

For example: If a package has a basic fare as Rs.3000 and the number of days as 15, then the package cost will be (3000*15), which is Rs. 45000.00. Since the number of days is 15, the discount percentage will be 7%. So, the discount will be (45000.0*(7/100)) which is Rs. 3150.00. Now, 12% of GST needs to be added. So the GST will be ((45000.0-3150.0)*(12/100)) which is Rs. 5022.00.

Therefore, the total cost for this package will be (((3000*15)-3150.0)+5022.0) which is Rs. 46872.00.

Develop as a menu-driven application. This application should show options

1. Add Package details
2. Display all package details
3. Search for a package with package id
4. Calculate package cost based on package id

Use layered architecture (Client (Main class), Service Layer, Dao Layer)

Classes to be created:

Model class - Package –attributes (packageid, source, destination, no_of_days and package cost)

PackageDao interface – declare the methods for adding package details, displaying all packages, calculating package cost and searching a package from the collection.

PackageDaoImpl class – implement the method declared in PackageDao interface

PackageService interface – declare the methods – addPackage, findPackageById ,fetchAllPackages , calculatePackageCost

PackageServiceImpl class –implement the methods of PackageService and call the methods of PackageDao class.

Main class – Create object of service class and call the methods by getting choice from the user.

Validation:

The packageId should be validated before calculating the package cost; only if the packageId is valid, the Package object should be added to the list. The packageId should be in the following format.

The packageId should contain exactly 7characters.

If the packageId is valid, then calculate the package cost, else throw a user defined Exception “InvalidPackageIdException” with a message "Invalid Package Id".

Com.main:

main.java:

```
package com.main;

import java.util.List;
import java.util.Scanner;
import com.model.Package;
import com.service.PackageService;
import com.service.PackageServiceImpl;
import com.exception.InvalidPackageIdException;
public class main {
  public static void main(String[] args) {
     Scanner sc = new Scanner(System.in);
     PackageService service = new PackageServiceImpl();
     while (true) {
        System.out.println("1. Add Package");
        System.out.println("2. Display All Packages");
        System.out.println("3. Search Package by ID");
        System.out.println("4. Calculate Package Cost by ID");
        System.out.println("5. Exit");
        System.out.print("Enter your choice: ");
        int ch = sc.nextInt();
        sc.nextLine();
        try {
           switch (ch) {
              case 1:
                 System.out.print("Enter Package ID: ");
                 String id = sc.nextLine();
                 System.out.print("Enter Source Place: ");
                 String src = sc.nextLine();
                 System.out.print("Enter Destination Place: ");
                 String dest = sc.nextLine();
                 System.out.print("Enter Number of Days: ");
                 int days = sc.nextInt();
                 System.out.print("Enter Basic Fare: ");
                 double fare = sc.nextDouble();
                 Package p = new Package(id, src, dest, days, fare);
                 service.addPackage(p);
                 System.out.println("Package Added Successfully!");
                 break;
              case 2:
                 List<Package> list = service.fetchAllPackages();
                 if (list.isEmpty()) {
                    System.out.println("No packages available.");
                 } else {
                    for (Package pack : list) {
                       System.out.println(pack);
                    }
```

```
                }
                break;
            case 3:
                System.out.print("Enter Package ID to search: ");
                String searchId = sc.nextLine();
                Package found = service.findPackageById(searchId);
                if (found != null) {
                    System.out.println("Package Found:\n" + found);
                } else {
                    System.out.println("Package not found!");
                }
                break;
            case 4:
                System.out.print("Enter Package ID to calculate cost: ");
                String costId = sc.nextLine();
                service.calculatePackageCost(costId);
                System.out.println("Package cost calculated!");
                break;
            case 5:
                System.out.println("Exiting...");
                sc.close();
                return;
            default:
                System.out.println("Invalid choice!");
          }
        }
        catch (InvalidPackageIdException e) {
            System.out.println("Error: " + e.getMessage());
        }
        catch (Exception e) {
            System.out.println("Something went wrong. Please try again.");
        }
      }
  }
}
```

Com.service:
PackageService.java:

```
package com.service;
import java.util.List;
import com.model.Package;
import com.exception.InvalidPackageIdException;
public interface PackageService {
  void addPackage(Package pkg) throws InvalidPackageIdException;
  List<Package> fetchAllPackages();
  Package findPackageById(String id);
```

```
  void calculatePackageCost(String id) throws InvalidPackageIdException;
}
```

PackageServiceImpl .java:

```
package com.service;

import java.util.List;
import com.dao.PackageDao;
import com.dao.PackageDaoImpl;
import com.exception.InvalidPackageIdException;
import com.model.Package;
public class PackageServiceImpl implements PackageService {
  private PackageDao dao = new PackageDaoImpl();
  public void addPackage(Package pkg) throws InvalidPackageIdException {
     if (pkg.getPackageId().length() != 7) {
        throw new InvalidPackageIdException("Invalid Package Id");
     }
     dao.addPackage(pkg);
  }
  public List<Package> fetchAllPackages() {
     return dao.getAllPackages();
  }
  public Package findPackageById(String id) {
     return dao.findById(id);
  }
  public void calculatePackageCost(String id) throws InvalidPackageIdException {
     Package pkg = dao.findById(id);
     if (pkg == null) {
        throw new InvalidPackageIdException("Invalid Package Id");
     }
     dao.calculatePackageCost(pkg);
  }
}
```

Com.dao:

PackageDao.java:

```
package com.dao;

import java.util.List;
import com.exception.InvalidPackageIdException;
import com.model.Package;
public interface PackageDao {
  void addPackage(Package pkg);
  List<Package> getAllPackages();
  Package findById(String packageId);
  void calculatePackageCost(Package pkg);
}
```

PackageDaoImpl.java:

```
package com.dao;

import java.util.ArrayList;

import java.util.List;
import com.model.Package;
public class PackageDaoImpl implements PackageDao {
  private List<Package> packageList = new ArrayList<>();
  public void addPackage(Package pkg) {
     packageList.add(pkg);
  }
  public List<Package> getAllPackages() {
     return packageList;
  }
  public Package findById(String packageId) {
     for (Package p : packageList) {
        if (p.getPackageId().equals(packageId)) {
           return p;
        }
     }
     return null;
  }
  public void calculatePackageCost(Package pkg){
          double basicFare = pkg.getBasicFare();
     int days = pkg.getNoOfDays();
     double total = basicFare * days;
     double discount = 0;
     if (days > 5 && days <= 8) {
        discount = total * 0.03;
     } else if (days > 8 && days <= 10) {
        discount = total * 0.05;
     } else if (days > 10) {
        discount = total * 0.07;
     }
     double afterDiscount = total - discount;
     double gst = afterDiscount * 0.12;
     double finalCost = afterDiscount + gst;
     pkg.setPackageCost(finalCost);
  }
}
```

Com.model:

Package.java:

```
package com.model;

public class Package {
  private String packageId;
  private String sourcePlace;
```

```
   private String destinationPlace;
   private int noOfDays;
   private double basicFare;
   private double packageCost;
   public Package(String packageId, String sourcePlace, String destinationPlace, int noOfDays, double
basicFare) {
      this.packageId = packageId;
      this.sourcePlace = sourcePlace;
      this.destinationPlace = destinationPlace;
      this.noOfDays = noOfDays;
      this.basicFare = basicFare;
   }
   public String getPackageId() {
      return packageId;
   }
   public String getSourcePlace() {
      return sourcePlace;
   }
   public String getDestinationPlace() {
      return destinationPlace;
   }
   public int getNoOfDays() {
      return noOfDays;
   }
   public double getBasicFare() {
      return basicFare;
   }
   public double getPackageCost() {
      return packageCost;
   }
   public void setPackageCost(double packageCost) {
      this.packageCost = packageCost;
   }
    public String toString() {
      return "Package ID: " + packageId +
            ", Source: " + sourcePlace +
            ", Destination: " + destinationPlace +
            ", Days: " + noOfDays +
            ", Basic Fare: " + basicFare +
            ", Total Cost: " + packageCost;
   }
}
```

Com.exception:

InvalidPackageIdException.java

```
package com.exception;

public class InvalidPackageIdException extends Exception {
  public InvalidPackageIdException(String message) {
     super(message);
  }
}
```

Output:

1. Add Package
2. Display All Packages
3. Search Package by ID
4. Calculate Package Cost by ID
5. Exit
Enter your choice: 1
Enter Package ID: 1000001
Enter Source Place: chn
Enter Destination Place: blg
Enter Number of Days: 35
Enter Basic Fare: 20
Package Added Successfully!
1. Add Package
2. Display All Packages
3. Search Package by ID
4. Calculate Package Cost by ID
5. Exit
Enter your choice: 2
Package ID: 1000001, Source: chn, Destination: blg, Days: 35, Basic Fare: 20.0, Total Cost: 0.0
1. Add Package
2. Display All Packages
3. Search Package by ID
4. Calculate Package Cost by ID
5. Exit
Enter your choice: 3
Enter Package ID to search: 1000001
Package Found:
Package ID: 1000001, Source: chn, Destination: blg, Days: 35, Basic Fare: 20.0, Total Cost: 0.0
1. Add Package
2. Display All Packages
3. Search Package by ID
4. Calculate Package Cost by ID
5. Exit
Enter your choice: 4

Enter Package ID to calculate cost: 1000001
Package cost calculated!
1. Add Package
2. Display All Packages
3. Search Package by ID
4. Calculate Package Cost by ID
5. Exit
Enter your choice: 2
Package ID: 1000001, Source: chn, Destination: blg, Days: 35, Basic Fare: 20.0, Total Cost: 729.12
1. Add Package
2. Display All Packages
3. Search Package by ID
4. Calculate Package Cost by ID
5. Exit
Enter your choice: 5
Exiting...