Java Bootcamp

Individual Application Exercise Specification Document

September 2022

Introduction

The following project is a basic version of a ticket purchasing and reporting application for an e-shop. Each academy participant is asked to implement the following project on their own. The programming language to be used is Java. All content in the application must be given in English.

For each of the steps discussed in this document, you are free to implement the requirements in any way you see suitable. You can decide on libraries, databases, logging, mechanisms for handling data, user input, user output, exceptions etc.

Keep in mind that there already exist best practices for building professional implementations. You are asked to build your software in a way that demonstrates your commitment to making the code maintainable, so make sure that your solution follows the best practices for interfaces, class definitions, packages, code quality and so on.

Contents

1. The Business Case

Assume that you are a developer for a travel company called **TravelCompany**, and the business analyst has given you a set of requirements that you need to implement for a new e-shop site.

For your convenience and for tracking purposes, requirements are logically grouped into steps. For every step/set of requirements, push your work to the corresponding GitHub repository by using one or more commits with proper comment(s).

All source code should be organized into Java packages. The proposed package nomenclature is **com.travelcompany.eshop**.

2. User Interaction

You may implement user interaction in any way you see suitable, but the simplest form of reading the console input and writing to the console output is sufficiently enough. You do not have to create a GUI or a web-browser interface.

3. Data modeling

The travel company has an extended clientele and offers a variety of itineraries in collaboration with inexpensive airline companies. The tickets are ordered and processed on-line. Example data are given in the appendix.

Tasks

Implement a console application to

- Design and implement the necessary domain classes to implement the core modeling of the system.
- Design the necessary enumerations, or any other supporting classes.
- Fill with demonstration data the appropriate collections.

4. Ordering and discount policy

Description

Based on the system's requirements, customers are categorized into **individual** and **business**. They can buy tickets by paying either with cash or by using a credit card.

You need to make sure that the system distinguishes between the purchase methods and customer categories, because the following discounts will apply upon the basic price when buying a ticket:

- Business customers get a discount of 10%
- Individual customers are subject to surcharge of 20% for all services
- There is a 10% discount when the customer pays by credit card
- There is no discount when the customer pays by cash

The discounts are cumulative. For example, a business customer purchasing a ticket using his/her credit card will receive a price reduction of 20% (10% as a business customer discount + 10% for paying with a credit card).

Tasks

- Design and implement the service layer by making use of one or more dedicated class(es). The proposed nomenclature is to include the token "**Service**" e.g., **CustomerService, TicketService**.
- Implement the purchase scenario by making use of the service class(es), the functionality implemented in the previous task and the domain classes.

5. Exception handling

Design custom exceptions that are thrown

- when creating a customer and the email of the customer is <whatever>@travelcompany.com
- when issuing a ticket , the requested itinerary does not exist and the given customer code does not exist
- when creating an itinerary and the airport code is not existing

Provide suitable code for exception handling.

6. Reporting

Description
Based on the purchases of each customer, the system must support the following reporting:

- List of the total number and list of the cost of tickets for all customers
- List of the total offered itineraries per destination and offered itineraries per departure
- List of the customers with the most tickets and with the largest cost of purchases
- List of the customers who have not purchased any tickets

The reports will be saved in memory and displayed to screen upon request.

7. Additional Functionalities

Description
The above requirements are sufficient for a successful submission. If you have time and want to enhance the operation of your system, you can add any type of functionality (within reason!) that adds value to the system.

Examples may include:

- Store the transactions of the customers when buying tickets with timestamps for the date and time of the purchases
- Have maximum number of tickets available for each Itinerary
- Save the collections to CSV files
- etc.

Be sure to provide documentation which explains the addons that you have implemented, otherwise they might go unnoticed.

8. Project delivery

The project will have to be published to a GitHub account of yours. Create a new private repository to push the project. In the class, you will be given a list of collaborators to add to the repository. DO NOT FORGET TO ADD ALL THE COLLABORATORS TO YOUR PROJECT.

In the GitHub repository your code will need to be ready to execute for anyone sharing the project. Create a README.md file which lists the requirements and the steps for the project to run.

The deadline for submitting your work is **Friday 16/9/2022 at 21:00.**

Your work will be evaluated for monitoring your progress and the deadline is an absolute business requirement – along with creating the code you will be evaluated in how well you can handle the date of submission for your work. GitHub keeps timestamps so ensure that all updates to your work happen within the deadline.

9. Appendix

The following are examples of how input and data may look like. Your code needs to use the same data format, but you can use a different data set if you want to extend your testing.

Customers

Id	Name	Email	Address	Nationality	Category
1	Maria Iordanou	miordanou@mail.com	Athens	Greek	Individual
2	Dimitriou Dimitrios	ddimitriou@mail.com	Athens	Greek	Individual
3	Ioannis Ioannou	iioannou@mail.com	Athens	Greek	Business
4	Antonio Molinari	amolinari@mail.com	Milan	Italian	Individual
5	Frederico Rossi	frossi@mail.com	Milan	Italian	Individual
6	Mario Conti	mconti@mail.com	Rome	Italian	Business
7	Nathan Martin	nmartin@mail.com	Lyon	French	Business
8	Enzo Collin	ecollin@mail.com	Lyon	French	Individual
9	Frederic Michel	fmichel@mail.com	Athens	French	Individual

Itineraries

Id	Departure Airport Code	Destination Airport Code	Departure date	Airline	Basic price
1	ATH	PAR	22/02/2022 13:35	SkyLines	300
2	ATH	LON	22/02/2022 13:40	SkyLines	420
3	ATH	AMS	22/02/2022 13:45	SkyLines	280
4	ATH	PAR	22/02/2022 14:20	SkyLines	310
5	ATH	DUB	22/02/2022 14:35	SkyLines	880
6	ATH	FRA	22/02/2022 14:55	SkyLines	380
7	ATH	FRA	22/02/2022 15:35	SkyLines	350
8	ATH	MEX	22/02/2022 16:00	SkyLines	1020
9	ATH	DUB	22/02/2022 16:35	SkyLines	770

Ordered tickets with payment (the payment amount gets calculated by the application)

Id	Passenger Id	Itinerary Id	Payment Method	Payment Amount
1	1	2	Cash	462
2	2	3	Cash	308
3	3	3	Credit Card	224
4	2	4	Credit Card	341
5	3	4	Cash	248

6	4	7	Credit Card	968
7	5	7	Credit Card	968
8	2	10	Cash	1122
9	1	3	Cash	308