16-4-2023

Requirements analysis Integrative Task II

Daron Mercado,
Juan José Barrera,
Alexis Jaramillo
ICESI UNIVERSITY

Client	Mercadolibre
User	Mercadolibre´s users
Functional requirements	The system must allow to: R1. Register product R2. Register order R3. Delete products R4. Increase product quantity R5. Search product R6. Search order R7. Filter numerical values R8. Filter by names R9. Select type of ordering R10. Save information R11. Manage exceptions
Context	Develop an online sales app for Mercadolibre that allows inventory management (add, delete, filter and search for products), in addition to having data persistence through JSON serialization to guarantee data integrity and correct handling of the exceptions.
Non-functional requirements	● Develop the project using TDD. ● Using a repository from day 1. ● Make at least 10 commits spread equi-temporally. ● For each of the 10 commits, report 3 simple quality indicators in the repository readme (bug-density, reliability, and completeness). The completion indicator is expected to be greater than 5 at the end of the product implementation.

Name or identifier	**R1. Register product**		
Summary	Allow the user to enter information about each product, including its name, description, price, quantity available, category, and number of times purchased.		
Inputs	Input name	Datatype	Selection or repetition condition
	productName	String	The program must throw an exception in case that the name already exists in the database.
	productDescription	String	No.
	productPrice	double	The program must throw an exception in case that the input does not correspond to a real number.

	productQuantity	int	The program must throw an exception in case that the input does not correspond to an integer.
	category	TypeCategory	The program must throw an exception in case that the input does not correspond to a Type category.
	purchasedTimes	int	The program must throw an exception in case that the input is negative or does not correspond to an integer.
General activities necessary to get the results	Once the user enters all the information of the product, the program will notify the user and it will save it in a database. the product will carry on staying there even though the application is closed.		
Result or postcondition	The product has been registered and saved in the database.		

Outputs	Output’s name	DataType	Selection or repetition condition
	message	String	No.

Name or identifier	**R2. Register order**		
Summary	Allow the user to record the orders made by users of the store, including information such as the name of the buyer, list of products, total price and date of purchase. When generating an order, the system must decrease the amount of available inventory.		
Inputs	Input name	Datatype	Selection or repetition condition
	buyersName	String	No.
	productList	String	The program must throw an exception in case that any product in the list

			does not appear in the product’s database.
	totalPrice	double	The program must throw an exception in case that the input is negative or does not correspond to a real number.
	purchaseDate	String	The program must throw an exception in case that the input does not correspond to a valid date format.
General activities necessary to get the results	Once the program receives all the information related with the order, it will notify the user and it will register it in a database. If the program is closed, the product will carry on staying there.		
Result or postcondition	The order has been registered.		
Outputs	Input name	DataType	Selection or repetition condition

	message	String	No.

Name or identifier	**R3. Delete products**		
Summary	Allow the user to delete products from the virtual store.		
Inputs	Input name	Tipo de dato	Condición de selección o repetición
	productName	String	The program must throw an exception in case that the product to delete does not appear in the database.
General activities necessary to get the results	Once the program receives the name of the product, it will use the binary search algorithm in order to find the product faster. If the system finds it, it will delete it from the database.		

Result or postcondition	The product has been deleted.		
Outputs	Input name	DataType	Selection or repetition condition
	message	String	No.

Name or identifier	**R4. Increase products quantity**		
Summary	Allow the user to increase the amount of each product already registered.		
Inputs	Input name	Datatype	Selection or repetition condition
	productName	String	The program must throw an exception in case that the product to increase the quantity does not appear in the database.

	quantityToIncrease	int	The program must throw an exception in case that the input is negative or does not correspond to an integer.
Actividades generales necesarias para obtener los resultados	Once the program receives the name of the product, it will use the binary search algorithm in order to find the product faster. If the system finds it, it will add the quantity entered by the user.		
Result or postcondition	The quantity of the product has increased.		
Outputs	Output name	DataType	Selection or repetition condition
	message	String	No.

Name or identifier	**R5. Search product**		
Summary	The product search engine should allow you to search for products by name, price, category and number of times purchased. Also the system has to implement a binary search algorithm to find matches.		
Inputs	Input name	Datatype	Selection or repetition condition
	productName (In case that the user's choice is searching by name)	String	The program must throw an exception in case that does not exist a product with that name in the database.
	productPrize (In case that the user's choice is searching by prize)	double	The program must throw an exception in case that does not exist a product with that prize in the database. The program also throws an exception if the input is negative or

			doesn’t correspond to a real number.
	productCategory (In case that the user’s choice is searching by category)	TypeCategory	The program must throw an exception in case that does not exist a product with that category or that category doesn’t exist in the database.
	numberOfTimesPurchased (In case that the user’s choice is searching by number of times purchased)	int	The program must throw an exception in case that does not exist a product with that quantity of times purchased in the database. The program also throws an exception if the input is negative or doesn’t correspond to an integer.
General activities necessary to get the results	Once the user chooses the option of “Search product”, the program will ask him/her the way of searching the product (by name, by prize…). After, the program will implement the binary search algorithm in order to get the product and its features.		

Result or postcondition	The product has been searched, found and shown by screen.		
Outputs	Output name	DataType	Selection or repetition condition
	message	String	No.

Name or identifier	**R6. Search order**		
Summary	The order finder allows the user to search by buyer's name, total price and date of purchase. Also the system has to implement a binary search algorithm to find matches.		
Inputs	Input name	Datatype	Selection or repetition condition
	buyerName (In case that the user’s choice is searching by buyer’s name)	String	The program must throw an exception in case that does not exist a buyer with that name in the database.

	totalPriceOfOrder (In case that the user’s choice is searching by total price of order)	double	The program must throw an exception in case that does not exist an order with that prize in the database. The program also throws an exception if the input is negative or doesn’t correspond to a real number.
	purchasingDateOrder (In case that the user’s choice is searching by total price of order)	String	The program must throw an exception in case that the input does not correspond to a valid date format.
General activities necessary to get the results	Once the user selects the option of “Search order”, the program will ask him/her the way of searching the product (by buyer’s name, by total price of order…). After the program will implement the binary search algorithm in order to get the order and its features.		
Result or postcondition	The order has been searched and found and shown by screen.		

Outputs	Output name	DataType	Selection or repetition condition
	message	String	No

Name or identifier	**R7. Filter numerical values**		
Summary	Allow range searches for numeric attributes (price, quantity available, number of times purchased). Also it can enter a maximum or minimum value to be searched.		
Inputs	Input name	Datatype	Selection or repetition condition
	filterCriteria==1	int	The program must throw an exception in case that the user enters a character or a negative number.
	numericValueTobeFilter	int	The program must throw an exception in case that the user enters a character

			or a negative number.
	minimumValue	int/double	The program will throw an exception if the input is negative or doesn’t correspond to a number.
	maximumValue	int/double	The program will throw an exception if the input is negative or doesn’t correspond to a number. The program will also throw an exception in case that the maximum value is less than the minimum.
General activities necessary to get the results	The user can find in the section of “Search product” and in the section of “Search order” an option (different to search by name, by price…) called filter. There, he/she can choose the searching criteria. (1.) if he/she wants to filter numeric values (2.) if he/she wants to filter string values (In this case the user chooses 1). After that, the user has to choose the option (1,2, 3...) that corresponds to the numeric value to be filtered (price, number times purchased…). Then, the program will use the inputs of		

	minimum and maximum value to filter the results of products or orders according to the case.		
Result or postcondition	The results have been filtered.		
Outputs	Output name	DataType	Selection or repetition condition
	valueFiltered	int/double	Provided that the filtering process has fulfilled all conditions

Name or identifier	**R8. Filter by string value**		
Summary	Allow interval searches for attributes of type string (name). Also the user can define a starting letter and an ending letter to search for products that are in that alphabetical range.		
Inputs	Input name	Datatype	Selection or repetition condition
	filterCriteria==2	int	The program must throw an exception

			in case that the user enters a character or a negative number.
	stringValueTobeFilter	int	The program must throw an exception in case that the user enters a character or a negative number.
	startingLetter	char	The program will throw an exception if the input does not correspond to a character.
	endingLetter	char	The program will throw an exception if the input is negative or doesn’t correspond to a number. The program will also throw an exception in case that the maximum value is less than the minimum.

General activities necessary to get the results	The following conditions must be met for the String value filter to work. The object class must be defined, a collection of objects must be created, and the filter methods must work correctly as long as the correct values are entered and exist in the system.		
Result or postcondition	Objects registered in the system that meet the valid filter requirements entered by the programme. Also, as a postcondition, the user may not enter a valid filter, and a warning of "Invalid character entered" will be displayed, and the user will be given the option to add another value.		
Outputs	Output name	DataType	Selection or repetition condition
	valueFiltered	String	Provided that the filtering process has fulfilled all conditions

Name or identifier	R9. Select sorting type.		
Summary	Before showing the results, the search engine must allow the user to select the order (ascending or descending) of the data displayed and the ordering variable.		
Input	Input name	Datatype	Selection or repetition condition
	sortingVariable	int	The system will throw an exception in case the input doesn't match any option or the input is different from type int.
	sortType	int	The system will throw an exception in case the input doesn't match any option or the input is different from type int.
General activities necessary to get the results	The user, after going through the different menus and before showing the result, should choose the variable to sort (represented by numbers). In case of being product		

	(name, price…) or in case of being order (buyerName, totalPriceOrder…). After that, the user has to choose the way which information is going to be ordered, ascendent (1) or descendent (2).		
Result or postcondition	Depending on the user's selection the program will use an ordering method, then it will be printed on the console the products ordered (taking into account the previous parameters).		
Outputs	Output name	DataType	Selection or repetition condition
	message	String	No.

Name or identifier	**R10. Save information**
Summary	The system must have data persistence through JSON serialization with the purpose of saving product information to files so the data is available even after the program is closed.

Inputs	Input name	Datatype	Selection or repetition condition
	Product	ArrayList	It is saved only if all class data are allowed inside the JSON object.
	Order	ArrayList	
	products	Product	It is stored only if all subdata of the class objects are allowed within the JSON object.
	orders	Order	
General activities necessary to get the results	The data entry must meet certain requirements for correct storage. This may include checking that the expected keys are present in the JSON, that the values are of the correct type, and that the program's data constraints are met.		
Result or postcondition	Objects created in the system by the user, stored in a JSON-like file.		
Outputs	StorageSystem	JSON	If all stored objects are allowed by JSON

Name or identifier	**R11. Manage exceptions**		
Summary	The system must handle exceptions to avoid unexpected states during product entry and search, as well as provide informative error messages to guide the user in case an unexpected error occurs.		
Inputs	Input name	Datatype	Selection or repetition condition
	invalidEntryInTheSystem	IOException	They are only executed when an exception occurs in the program caused by a client error in operation.
	selectOnTheMenuNotAvailable	IOException	
	filterEnteredNotAllowed	IOException	

General activities necessary to get the results	It is necessary that a method created to fulfill a customer need in the program is being executed, then the customer makes a mistake and enters an option not allowed in the system.		
Result or postcondition	Notice to the customer of an error in the system, and the option to re-enter the option he/she wishes to request.		
Outputs	message	String	Only if an exception occurs in the program.