Estudiantes: Juan Pablo Herrera Herrera

Juan sebastián moreno ruiz

Changes made to the Project

UpdateQuality method refactoring

We decided to refactor the updateQuality method of the ItemService class, since we considered that it was not readable, in addition to improving its extensibility and maintainability considering how unintuitive it was.

Item class changes

To refactor the updateQuality method, we decided to change the model of how the Item class was proposed. Changing this for a parent class, which represents the normal type items and from this the other types of items AGED, LEGENDARY and TICKETS will inherit. In addition, due to encapsulation and inheritance, we decided that each of the 4 mentioned classes would have the updateQuality method implemented so that it is applied depending on the type of Item.

Within the Item class a template Method was created using the template Method pattern a modification of this pattern was created the updateQuality method was created within item so that each class is responsible for overwriting the way its attribute is updated quality, using a hook method within this method to override the way the new quality value is calculated each time.

Also, a method called ensureQualityBoundaries was added to increase readability within the updateQuality method.

ItemFactory class changes

We also created a class called ItemFactory, based on the single responsibility principle, which establishes that each class must have a single responsibility within our software, and this responsibility must be defined and concrete. Therefore, to make the changes to the model of the new item instances, the ItemFactory class was created. In ItemFactory 2 were implemented:

Methods newTypedInstance: Which according to the type of item creates a new instance of a specified subclass depending on the type of the item.

newBaseItem method: Which creates base items of the parent class

Inside ItemFactory a private constructor was created to not allow instantiating the class

Constants class changes

In addition to everything mentioned above, to improve readability and increase the descriptiveness of the code, we decided to create a class called Constants inside the utils folder to add all the temporary variables used inside the new changes.

Responsible for creating new item instances thus decoupling the model class from the responsibility of its own creation