Project Documentation

### Team name: Rusty The Clown

### Team members: Vasil Vasilev - v.a.vassilev

### Boyan Boyanov - Boyan\_Byanov

### Maria Popova - mariq88

### Daniela Vassileva - Daniera

### Desislava Dimitrova - dpavlinova

### TFS URL: https://dpavlinova.visualstudio.com/DefaultCollection/StoreKrustyTheClown

### Project purpose:

This is a software application serving the need of a store with capablities to manage the different products, customers and their order history. It is build using the best practices of object oriented programming and design. The domain managed by the application has been represented with a class hierarchy, shown in the diagram 1.1.

The main class in the application is Store. It contains a list of the abstract class Product, which is the base class for the different product types. The inheriting classes are Book, Movie, Music, StudentAccessory, Toy. The Product class also implements the interfaces IRentable, Idicountable and ISearchable. The IRentable interface provides functionality for checking whether an item can be rented through the IsRentable method. The IDiscountable interface provides functionality for getting product discounts. The ISearchable interface provides functionality for searching product by given name or by given author name. There is also a class Client which contains some information about the clients – name and ID number as well as the shopping card (the current purchase amount) and the client “History”, i.e. previous purchases. The class Client inherits the interface IDiscountable. The implementation of this interface is as follows. If the current purchase price is over 200 and previously the client has bought something and the last purchase was within the last 3 months, give 20 % discount over the current purchase. If this is not the case, then if the current purchase price is over 500 and the previous purchase amount is over 500, give 10 % discount over the current price. If the above conditions are not satisfied, then the client cannot get any discount.

The structure History, used inside the Client class contains the information for the previous client purchases. It has 2 properties – purchase and purchase date and time; The property purchase is of type Product and has properties - bought product, its price and quantity and the name of the item which was bought.

For the purpose of logging exceptional situations warning and information has been created a class called Logger. It implements the Singleton Design Pattern. The idea of a Singleton Pattern is to have only one available instance of a class. There for the constructor is set to private and there is a static method in the class, that either creates a new instance of a class or returns an existing one if such has been created. In this class there is a method Log**,** which writes the type of error and the passed message to a text file. The error types are taken from the ErrorTypes enumeration.

The Store class has a method CheckAvailability, which checks whether product with a specified name exists. If the product doesn’t exist the method throws a custom exception ProductNotExistException. The above method is called in a try catch block, which catches the exception and logs it with our logger.

In the Product class we have declared two events. The first one is called AlmostEmptyInventory and is rised when the customer trys to order a product and the quntity is less than 5. The second one is called EmptyInventory and is rised when the quantity of a certain item is zero. These events are included in SellOneQuantity method, so we check what is going on only when a customer requires to buy some product.

In case his buy request leads to change in quality, required in any of the two events, the appropriate one fires. The subscribing methods are:

* The delegate of EmptyInventory method subscribes to EmptyInventory Event.
* The delegate of AlmostEmptyInventory method subscribes to AlmostEmptyInventory Event.