<Assignment A>

Student: Bulea Ioana - Florina

**Group: 931**

# Requirements Analysis

## Assignment Specification

The ReNew system is required to provide it’s users with a list of various products for home improvement and handles the renovation process. So once a user is signed in he is presented with the options to create a new project and choose the products he wants to use. The end goal is to obtain a list of products to be purchased (a shopping list) and a summary of the renovation process (installation summary).

Every product has a price, color and unit of measure and is tied to a specific install method. An install method describes how the product is installed and what additional items you need (glue, nails etc.) it has a name and description and a price for the work required to be performed.

The shopping list should contain a list of products and associated items based on install method for a given project. It should display the price computed for every line item and a total price.

The installation summary should contain a list of install methods needed for a given project. It should display the price computed for every line item and a total price.

# Use-Case Model



**Use-Case description**:

Use case: Log In

Level: sub-function

Primary actor: Client

Main success scenario: The Client is presented with the Log in panel. They introduce their username and password and Click on the “Enter” button to begin using the application.

Extensions: In case the username or password (or both) is not valid, the application will display an error message

Use case: Add a product to shopping list

Level: user-goal level

Primary actor: Client

Main success scenario: The client selects at least a product from the product table and presses the “Add” button. The selected products will be added to the shopping list, along with the items necessary for the installation method.

Extensions: If the “Add” button is pressed before a product is selected, the application will display an error message.

Use case: Remove a product from shopping list

Level: user-goal level

Primary actor: Client

Main success scenario: The client selects at least a product from the shopping list and presses the “Remove” button. The selected products will be removed from the shopping list, along with the items necessary for the installation method.

Extensions: If the “Remove” button is pressed before a product is selected, the application will display an error message

Use case: View necessary items

Level: user-goal level

Primary actor: Client

Main success scenario: The Client presses the “View Installation Summary” button and a window containing the installation summary of all the products in the shopping list is displayed. This contains all the methods of installation, their description, the necessary items and their prices.

Extensions: If the shopping list is empty, the installation summary will be empty too.

# Design

The application is a Windows Forms application, created using the C# programming language.

There are two packages of classes (Model and Repository), along the Form1, Necessities and Program classes. The Model Package includes the Product, User and Install\_Method classes.

The Repository Package contains an interface and three classes: Install\_Repo, User\_Repo and Shop\_Repo. Install\_Repo and Shop\_Repo implement the IRepository interface.

Install\_Repo contains a list of all the installation methods of the products selected by the user.

User\_Repo is a list of all the registered users of the application, alongside their passwords.

Finally Shop\_Repo is a list of all the products selected by the client in the application.

The Form1 class contains the actual implementation of the GUI of this application. It contains instances of all the Repository objects and also of the class Necessities, alongside lists of products and installation methods.

## Class Diagram



## 3.2 Patterns

I decided to use Abstract factory pattern in order to make a more reusable code. To do that, I made an interface Item for the Product and Install\_Method models and an abstract class, named AbstractFactory that has a method which returns an Item of type Product or Install\_Method, depending on the parameters it receives. The abstract class AbstractFactory is extended by the ItemFactory class.