# Assignment 6: USP

## Background:

*PureFoods*, an up-and-becoming chain of grocery stores, has heard about the *EZShop* app you designed and is interested in hiring you to further explore the possibility of building both the app and the payment system for it (the one used by the cashiers). The managers in charge of the project, Brad and Janet, studied computer science in college and were fascinated by UML, so they want you to follow a UML-based process for this project.

## Goals:

- Perform a slightly simplified version of the Inception and Elaboration phases of a Unified Software Process aimed at building an extended version of the system you designed in the previous assignment.
- More generally, get experience with (part of) the Unified Software Process.

## Requirements (repeated here for your convenience):

1. You can assume that each existing customer already has a card with a QR code that can be scanned by the *EZShop* app and encodes the following customer information:
    a. First name
    b. Last name
    c. Zip code
    d. Email address
2. Similarly, all products in the supermarket are suitably labeled with a QR code that can be scanned by the app and encodes the item's ID and price.
3. Alcoholic beverages are treated differently from other products, as a special tax applies to them. To indicate that, their ID always starts with an "AB" prefix.
4. Before starting to grocery shop, a customer would launch the EZShop app. When started, the app would:
    a. Launch the QR code scanner
    b. Read the customer card's QR code
    c. Suitably initialize the customer's information based on the information read
    d. Initialize an empty cart with an empty list of items and coupons
    e. Go into grocery shopping mode
5. While in grocery shopping mode, a customer can then use the app to add an item to his/her list of currently purchased items before putting them in the physical cart. To do so, the customer must press the "+" button on the app's screen. At that point, the app:
    a. Launches the QR code scanner
    b. Reads the QR code of the item, which provides the item's ID and price

    c. Adds the item's ID and corresponding price to the list of current items in the customer's cart
6. While in grocery shopping mode, a customer can also scan coupons for a given product. In that case, the app:
    a. Launches the QR code scanner
    b. Reads the QR code of the coupon, which provides the product's ID and the corresponding discount (assume a simple coupon model, in which a coupon contains a discount that simply applies to each item with that ID)
    c. Adds the product's ID and corresponding discount to the list of current coupons in the customer's cart
7. If a customer makes a mistake or changes his/her mind about an item, he/she can remove the item by pressing the "-" button on the app's screen. At that point, the app:
    a. Launches the QR code scanner
    b. Reads the QR code of the item, which provides the item's ID and price
    c. Removes an entry corresponding to that ID from the list of current items in the customer's cart
8. When done, a customer can press the "Pay" button on the app's screen, which generates a QR code that encodes (1) the customer's information, (2) the items and coupons lists, and (3) the total value of the cart, computed taking into account the items in the items list (including whether they are alcoholic beverages or not) and the coupons in the coupons list. The customers would then show the QR code to a cashier, who would scan it, possibly double check the items list with the items physically in the cart (based on a random selection meant to keep customers honest), collect a payment method from the customer, and conclude the transaction when the payment is successfully completed, which would also send an email to the customer with their receipt.

# Deliverables:

1. Use-case model (see template). Make sure that your use cases, and corresponding descriptions, cover the most relevant scenarios of usage of the app/system.
2. Supplementary requirements (i.e., requirements that do not fit the use-case model, such as non-functional requirements). These can be provided as a simple list in a document called `suppl-requirements.{md|pdf|txt}`.
3. Design document (see template). For the part of the class diagram related to the *EZShop* app, you can reuse the design you created for the previous assignment (or an improved version of it).

# Notes (important–make sure to read carefully):

1. For details about the Inception and Elaboration phases, and about USP in general, see the corresponding lessons.
2. In preparing your deliverables, make sure to consider **both the app and the payment system**.

3. You do not have to develop the actual app or the payment system; that is, **you do not need to write code for this assignment**.
4. In designing the system, you can assume that–just like QR scanning–payment processing and email capabilities can be provided by libraries, through utility classes.
5. If you identify incompleteness or inconsistency issues in the requirements, feel free to either ask for clarifications or make explicit assumptions and proceed accordingly (**if so, be sure to document these assumptions in the supplementary requirements file**).
6. Put all the files for this assignment in directory "`Assignment6`" in the repo we assigned to you (as usual).
7. All documents must be in markdown (preferred), PDF, or plain text format (i.e., a `.txt` file).
8. As usual, after committing and pushing your changes to GitHub, submit on T-Square the corresponding commit ID.