## 

## Code Repo

<https://github.com/richardcwalker/katacheckoutscanner>

## To Do

* DAL – Need DB/EF context etc
* DI – Remove constructor ‘new up’ with some framework (Structure Map etc)
* Error Logger to log errors to DB + log .Net errors as well
* Use Azure as storage for amounts and transactions

## Assumptions

A transaction ID would be passed from the scanner software along with the SKU.

## Spec

A supermarket requires a working checkout. MVP is to scan products **and periodically ask for a total price**, considering any special offers that apply to the product.

Items:

|  |  |
| --- | --- |
| **SKU** | **Unit Price** |
| **A99** | **0.50** |
| B15 | 0.30 |
| C40 | 0.60 |

Special Offers:

|  |  |  |
| --- | --- | --- |
| **SKU** | **Quantity** | **Offer Price** |
| **A99** | **3** | **1.30** |
| B15 | 2 | 0.45 |

The checkout accepts items scanned in any order, so that if we scan a pack of Biscuits (B15), an apple (A99) and another pack of biscuits, we’ll recognise two packs of biscuits and apply the discount of 2 for 45. **Prove your solution works for this scenario**.

Please push your work to a remote git repository (e.g. GitHub). Commit as you go to show your working process, rather than just one big commit at the end.

Work your way through this checklist:

* Create a new solution
  + Include a class library for the logic
  + Include a test library for unit tests (feel free to use whatever test library you are most comfortable with)
* **Prove you can scan an item at a checkout (AddScannedItem method in ItemService)**
* **Prove you can request the total price (GetTotalPriceOfItems method in ItemService)**
* Prove unknown sku are dealt with
* Introduce special offers
  + Amend your prior implementation to consider offers on items
* Prove you can request the total price inclusive of offers

This kata covers just the middleware, do not be concerned with UI or data access.

No one solution is “correct”,

* Your process
* Testable code
* Maintainable code
* Abstraction at sensible points
* How you would refactor your solution to deliver future requirements