Edward Demkowicz-Duffy

Crossley Heath School  Halifax

Computer Science COursework 2017

# Analysis

## Brief

My application will be built for a client who produces novelty clocks and coasters. My client presently uses multiple methods for selling their products; the primary four are the two online storefronts eBay and Etsy, the client’s personal website, and a large physical stall held at Christmas in Manchester. The clocks are produced with the central sections of vinyl records, the coasters are made from custom made plastic bases and have (custom made) stickers applied to the top. They are sold in tins of 6, with the tin and the sticker to go on the tin being two separate components too. The client keeps a central inventory which the online storefronts draw from, and takes a different inventory with him each day of the Christmas market.

I plan to use ASP.NET with C# to develop the website and the code behind it, and a Microsoft Access database to manage the data.

## Project Parameters

My goals for the application according to the client’s needs are:

* Keep track of all products in inventory
* Keep track of total sales from all online storefronts via API integration
* Keep track of all user interaction with the application via logging to a text file
* Keep the storefronts up to date with current stock
* Keep track of costs and gross & net profits
* Give a detailed breakdown of sales statistics for the online storefronts (including visual representations) on:
  + A time period divided basis including options for months, days and years
  + Comparisons of different selling statistics:
    - Sales by user
    - Sales by storefront
    - Sales by date/month/season
    - Predict the most popular times of year and outlets for sales from extrapolation of visible data
* Allow the user to add sales manually for physical “face to face” sales, including those at aforementioned Christmas market
* Allow the client to print a customizable catalogue to take with them to the market to display
* Have an authentication system, ie. a user logon page before the user can access sensitive information or be able to edit anything and allow different ID/password combos to be created for other employees with varying permissions level, for example the admin account could remove and change products and sales data, and a employee account could only view these items
* Provide a web-based UI for all these functions with intuitively and clearly displayed information and interactive elements; including but not limited to graphs for statistics with provision for sorting data and the ability to “zoom in” to view individual data and compare specific points

## Storefront Synchronisation

There are three online, trackable locations where the products are sold: eBay, Etsy and the client’s personal website. These three outlets are different in significant ways. The personal website does not show stock information other than marking products as “out of stock”. Etsy and eBay keep track of stock by showing how much of an individual product is left. Both sites provide publicly accessible APIs with sandbox modes to allow the user to experiment and debug their software.

For Ebay, I plan to use their “inventory API” to keep the stock pages up to date on how much stock remains according to the data in the project’s database. I will use their “analytics API” to gather information to display alongside the analytic data I generate myself, and the “fulfilment API” to manage shipping. All their APIs use HTML requests to communicate with the remote application and Oauth to verify its identity. Oauth is an open standard dictating methods of verification using identity tokens generated and exchanged between the two communicating parties, in this case my application and the Ebay servers.

Etsy likewise uses HTML requests and oauth for third party apps to integrate with its services so my practice will be much the same; however, their API is not so well divided. They offer their “sellers API” which allows the user to create listings and manage product shipping.

The client’s public website, as mentioned, only displays whether the product is out of stock or not, so this task is relatively simple. I plan to develop an application in the form of a “plugin” for the server hosting the client’s site in either JavaScript or C# which can change the value on the page, and allow my main program to communicate with that under appropriate security. This I will achieve by generating a token which the “applet” will recognise and hashing it, then sending it to the applet which should recognise it and accept it.

## Data Management and Storage

I will use a relational database in Microsoft Access to store all the relevant data. There will be five tables:

* The user credentials table, which will contain userIDs, hashed passwords, usernames and “clearance levels” which the program will check against when users try to login, and will also use to label actions which are logged.
* The products table, which will contain generic entries for all products to be referenced when orders/purchases are made. The fields will be the product’s display name (correctly formatted in English), the ID, the product’s reference name (which will be formatted in camel case for the sake of development consistency), the product’s typical price, the type or product (ie. coaster or clock) and the product’s stock level.
* The customers table, which will be a record of all the known users that have made purchases from the online outlets. This will contain their online username and online identifier string if applicable, and their local ID. This will allow me to credit portions of sales to individual buyers.
* The orders table, which will be a record of every individual purchase made. It will contain the ID of the product purchased as a relational link, the gross profit from the sale, the sale ID itself, the local ID of the customer if applicable (if bought from ebay or etsy) as a relational link and the volume of the product sold.

## Notes

-Things I did in development:

* Tried using pure code to handle database but this didn’t work so I used datasets
* Got a custom SQL sanitization class working
* Tried using HTML verification instead of sessions
* Made a custom logging class for development and debugging

## Initial Notes (Please ignore)

Stock:

-Records to be turned into clocks

-Stickers

-Blank Coasters

-Completed coasters (# of sets (6 in a set))

-Clocks – limited selection of records which are ordered frequently:

-Tins

-Costs of stock?

Core features:

-Separate mode for Christmas market or other special occasion:

-Separate inventory

-Printable catalogue

-Inventory management:

-Overview of completed stock

-Separate screen for total inventory/total components?

-Developmental features “add x amount of y”

-Tables:

-Orders

-Deliveries of components

-Production of completed products

-Clients

-Inventory

Extra features:

-Interface with online outlet’s APIs

-Centralize stock values/inventory across all outlets

-Authentication system

-Infographics

-Product comparisons and other data aggregation

Advantages:

-Volume sold; profit and gross

Notes:

-Postage is paid by client

[phil@philfowler.co.uk](mailto:phil@philfowler.co.uk)