A-LEVEL COMPUTER SCIENCE PROJECT

Inventory management system

OCR A-Level Computer Science

Townley Grammar School

Arun Patel

Table of Contents

[Analysis 3](#_Toc475520603)

[1 Problem identification 3](#_Toc475520604)

[1.1 Description of the organisation 3](#_Toc475520605)

[1.2 Description of the problem 3](#_Toc475520606)

[2 Stakeholders 4](#_Toc475520607)

[2.1 Stakeholders who will have an interest in the solution 4](#_Toc475520608)

[3 Research the problem 4](#_Toc475520609)

[3.1 Analysing the problem 4](#_Toc475520610)

[3.2 Existing solutions 6](#_Toc475520611)

[3.3 Advantages and disadvantages of research 8](#_Toc475520612)

[3.4 Forms of data 9](#_Toc475520613)

[4 Data collection 11](#_Toc475520614)

[4.1 Questionnaire 11](#_Toc475520615)

[4.2 Interview 14](#_Toc475520616)

[5 Diagrams 16](#_Toc475520617)

[5.1 The current system 16](#_Toc475520618)

[5.2 Capabilities and Limitations of the current system 17](#_Toc475520619)

[6 The proposed solution 18](#_Toc475520620)

[6.1 Solution requirements 18](#_Toc475520621)

[6.2 Summary of the solution to be created 20](#_Toc475520622)

[7 Signatures 21](#_Toc475520623)

[Design 22](#_Toc475520624)

[1 System objectives 22](#_Toc475520625)

[1.1 User interface 22](#_Toc475520626)

[1.2 Input 22](#_Toc475520627)

[1.3 Processing 22](#_Toc475520628)

[1.4 Output 22](#_Toc475520629)

[1.5 Internal Objectives 23](#_Toc475520630)

[2 Scheduling 23](#_Toc475520631)

[2.1 Design plan 23](#_Toc475520632)

[2.2 Development plan 24](#_Toc475520633)

[3 New System 25](#_Toc475520634)

[3.1 Use-Case diagram of new system features 25](#_Toc475520635)

[3.2 Input Output diagrams 27](#_Toc475520636)

[3.3 Inventory Database ERD 28](#_Toc475520637)

[3.4 Data Dictionary 29](#_Toc475520638)

[3.5 Database normalisation 31](#_Toc475520639)

[4 User Interface 32](#_Toc475520640)

[4.1 Splash screen design 32](#_Toc475520641)

[4.2 Login screen design 32](#_Toc475520642)

[4.3 Main menu 33](#_Toc475520643)

[4.4 Inventory page 34](#_Toc475520644)

[4.5 For sale page 35](#_Toc475520645)

[4.6 Performance page 38](#_Toc475520646)

[4.7 Smart Phone login screen design 40](#_Toc475520647)

[4.8 Smart Phone stock replenishment screen 41](#_Toc475520648)

[5 Data capture 42](#_Toc475520649)

[5.1 Validation 42](#_Toc475520650)

[5.2 CRUD permissions 44](#_Toc475520651)

[6 Algorithms 46](#_Toc475520652)

[6.1 Structure diagram 46](#_Toc475520653)

[6.2 Class diagram 48](#_Toc475520654)

[6.3 Algorithm design 49](#_Toc475520655)

[6.4 Database Queries 54](#_Toc475520656)

[7 Test Strategy 56](#_Toc475520657)

[7.1 Test plan 57](#_Toc475520658)

[8 Signatures 60](#_Toc475520659)

[Development 61](#_Toc475520660)

[References 62](#_Toc475520661)

[Table of figures 63](#_Toc475520662)

# Analysis

## 1 Problem identification

### 1.1 Description of the organisation

Best-One New Cross, is a convenience store selling everyday items such as food and drink to locals in the area. Having been established for more than 35 years, it has a well-established and loyal customer base all with different needs. When the shop was first opened, it featured a very small range of products, suited for a small audience with similar needs, however, as more and more customers, who had different needs, started to visit the shop, it needed to expand its product range to satisfy customers.

Best-One is a chain of convenience stores based in the UK and Jersey. The company has over 600 stores throughout the UK, with most of its stock being sourced by Best way cash and carry. Best-One New-cross was opened and ran by the current owner’s father as a franchisee of the Best-One franchise. Once the current owner, Purnesh Patel, was of a suitable age and had enough experience, the shop was then passed on to him. As the shop grew and grew, the decision was made to buy the building next to it, allowing for more space for expansion

### 1.2 Description of the problem

When new stock is brought into the shop, a member of staff writes down the name of the item onto a piece of paper, which is stored in a filing cabinet. When stock is taken out of the storage room and put on display, the member of staff must find the item on the list and cross it out. This can be immensely difficult as there is often hundreds of items on the list all with similar names. It has also led to errors in the past when staff members cross out the wrong item. It is almost near impossible to work out who made the mistake since there is currently no way of tracking who made the changes. This can lead to blame being passed around to the wrong employees which can severely de-motivate the staff member. The prices of some items can change often, when this happens it can cause confusion as the prices of items stored in the system don’t match others.

These problems can be solved by making use of computational methods. For example, abstraction can be used during the design of the interface, by not including unnecessary information. Instead of asking the user where the item came from, it can be suitable to ask the user the name of the item and the quantity of that item.

Furthermore, decomposition can be used to solve problem of working out who made an error when managing stock. For example, by decomposing the problem, I can understand that everyone who alters the inventory has a name. They will also do this at a specific time. From this, I can come up with a solution to this problem. By implementing a security system, the solution can detect who is making changes to the inventory. In addition, the solution can also log the time of day that the staff member altered the inventory. By doing this, I can record who has made changes to the inventory and at what time.

As searching for items is a huge issue for the organisation and often leads to errors. Queries can be implemented into the application that allows the program to search the database holding all the items for the desired record. Methods of computational thinking such as thinking ahead can be used here in the form of caching. Frequently searched for items can be cached, so that the user can quickly go to the items details without having to waste time. Indexing can also be implemented into data stored in the system, this allows the system to search for data quicker, making queries on large pieces of data take a shorter amount of time. Indexing involves putting markers into sections of the list of data to indicate the data underneath are all of a same category.

Automation can be used when determining which food products have gone out of date. Rather than the user have to analyse a list of products and find which products have gone out of data , which can lead to errors such as missing an out of date item, the solution could feature an automated alert to the user , making them aware of products that have gone past their expiry date.

Data collection allows me to take in information from the client. This could be in the form of questionnaires and interviews. I could question my client on what the solution to their problem should include and be able to do, ensuring the final solution is correct. This can be used together with data analysis, to make sense of the data I have collected, this could be achieved by making use of graphs and charts.

## 2 Stakeholders

### 2.1 Stakeholders who will have an interest in the solution

The main stakeholder of this solution is my client, Purnesh Patel. This is the person who has asked to create a solution to a problem that they have identified. This person will make the most use solution, therefor the solution must be suitable for this person. Since this stakeholder has limited experience in computer systems and using software, it is essential that the software is as user friendly as possible, which means that I can make use of a Graphical User Interface (GUI). A GUI has features such as windows, icons, menus and pointers, which can make software much easier for the user to use, especially since they have limited experience. The solution will also require the user to input a lot of data, therefor it is essential for the system to make use of validation. Validation is an automatic computer check to ensure that the data entered is sensible and reasonable. It does not check the accuracy of data.

Another stakeholder of the solution, is the members of staff at the shop. They could potentially make use of the system as much as the main client. This stakeholder will act as the audience of the solution, the audience is the person who will look at the finished product. Since some members of staff have a lot of experience in using Computer systems and others have little experience, I will have to design the solution in such a way that can be suitable for both kinds of audiences. For example, creating an interface with a lot of help dialogs and tooltips can make it very frustrating for skilled users to use the solution, whilst removing all help features from the solution can make it very hard and even impossible for inexperienced users to use the solution.

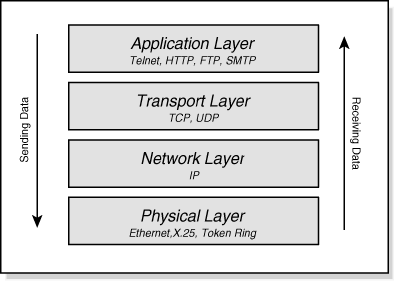
## 3 Research the problem

### 3.1 Analysing the problem

The problem which I am creating a solution to must satisfy the end user of the system. For example my client wanted the solution to be useable on different computers, where the data accessed and saved on one computer is available on another. To do this, I could make use of networking, saving the data in a small server. I could also achieve this by making use of real-time databases.

Google’s Firebase is an example of a service that offers real time databases. It is made for web based solutions but has third party wrappers for desktop programming languages such as C#. Real time databases allows each instance of the solution to be able to create, read, update and delete records in real time, so changes made by one instance of the solution will be seen by another. Initially Firebase can be used at no additional cost, however, in order to expand storage capacity and speeds you must pay a fee in order to upgrade to a higher package. This would be necessary if the solution is going to experience high traffic loads. This is unlikely since the solution is only going to be used by the client and staff members, which would not cause any delays due to traffic. If the store was to expand, offering more and more products, the Firebase package may have to be upgraded in order to expand the capacity of the database.

In order to access all the data of the system, the user must be using a computer connected to the internet, this creates a software and hardware requirement – access to the internet. This also increases the cost of the overall solution as fees would needed to be paid in order to keep the server running without stopping. To implement this feature I can make use of a computational technique, layering. “Layering is the organization of programming into separate functional components that interact in some sequential and hierarchical way, with each layer usually having an interface only to the layer above it and the layer below it”. In order to achieve layering, computational thinking methods such as decomposing the problem and abstraction must be used. Hardware such as a router are needed in order to access the internet. Furthermore, a subscription with an Internet Service Provider is needed, this can be very costly. If the internet has slow upload and download speeds the user could experience lag whilst using the solution.



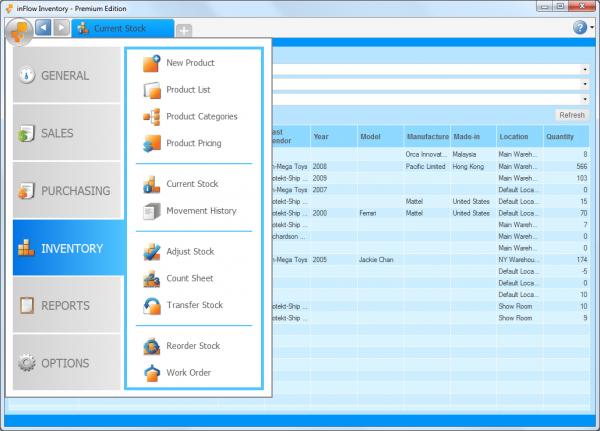
(Fletcher, 2016)

### 3.2 Existing solutions Acctivate Inventory Management - Product specifications

(Anonymous, 2016)

Acctivate inventory management software is a potential solution to the proposed problem. It is an off the shelf product and does not offer bespoke features, tailored to the client. Based on the features Acctivate has to offer, I could include some of fields shown above when storing the information of a product. For example, I could also allow the user to assign a product to a category. The advantage of doing this, is faster searching time, as indexing can be used. Rather than the solution searching the entire list of products it can search for products in the chosen category, therefore reducing the searching time. Also, I can see that Acctivate gives each product an ID, this could be a feature of the final solution, since all the products will be stored in a database I can make use of IDs to give each product a unique identifier, making duplicates unique from each other.

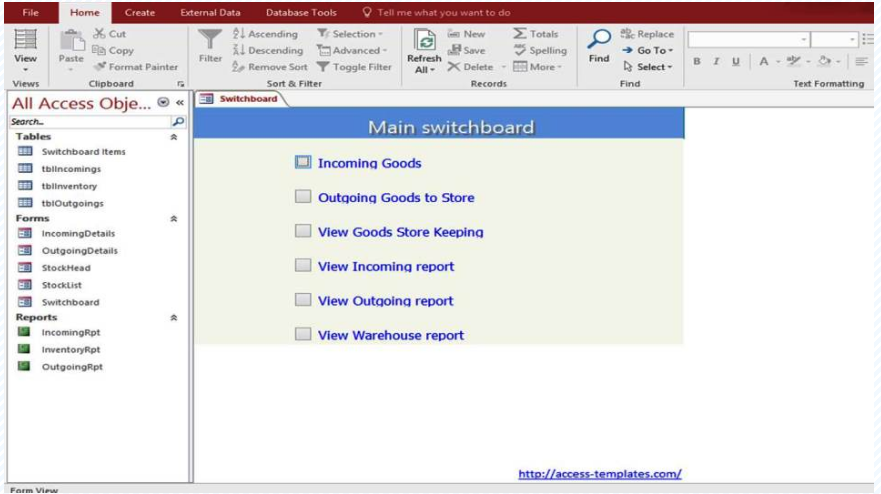
On the other hand, the product shown above has a cluttered user interface, which can come across as overwhelming for inexperienced users. For example it features many tabs near the bottom of the page, this can come across as overwhelming. It also allows the user to classify the product into different categories such as item type, product type and product class. This allows data to be stored in a more organised fashion allowing for quicker searching times but can be very time consuming for the user.

(Anonymous, 2016)

InFlow inventory is another potential solution to the problem. This program has a more de-cluttered interface, making it more suitable for inexperienced users. Features such as tabs, tables and menus are all features that I could include in my solution. The use of large menus on the left hand side could be beneficial to inexperienced as the icons and text are large and easy to view.

Rather than presenting all of the data entry fields on one interface like Acctivate inventory does, InFlow makes use of large icons and text, shown on the left hand side of the image. When the user clicks on one of these icons, more options appear. This is known as a menu driven interface, which is capable of displaying many options to the user whilst keeping the main user interface tidy, this can lead to the user feeling less amounts of stress due to being overwhelmed.

In the top right hand side corner, the interface contains a question mark. If the user clicked this button when using Inflow Inventory, they would be taken to a help dialogue, where they can find information on how to solve a problem they might have. This could be a very useful feature to include in the solution, since the users of my solution are inexperienced, they are likely to need additional help to use the solution, and this could be one way in which the solution could provide this.



(Templates, 2017)

Another solution to the problem would be to use Microsoft Access database software. Despite the solution being much easier to develop, it would be a very basic inventory management solution. It would not be able to incorporate bespoke features such as Artificial Intelligence and machine learning.

Shown in the image above is an example of an inventory management solution created in Microsoft Access. The main menu form is shown in the image it is clear to see that the user interface is very basic, which can be beneficial is inexperienced users. However, this means that there is not much room for customization, which means that developing a bespoke product in excel can be very difficult. For example, some user interface features that the client may want, may be unavailable in Microsoft Access.

Furthermore by making user Microsoft Access, the user is required to have the software installed on their machine, which can further increase costs. In addition, it also means that a mobile based solution cannot be created, if Access is only being used.

### 3.3 Advantages and disadvantages of research

The research I have made on existing solutions has allowed me to gain a better understanding of what is required of an inventory management system such as it being able to perform CRUD (Create Read, Update and Delete) operations on the database. It has also showed me how to create an effective User interface, by showing me poor and good designs. I now have a better chance of creating an interface more suitable for my client.

However, my research has not showed me how an inventory works exactly for a small business such as a convenience store. They have all been based on large businesses with a large inventory. Although the principle will be the same, there will be subtle differences between the system I create and the systems I have researched.

### 3.4 Forms of data

#### 3.4.1 Paper based

Currently, there are two paper documents in use. One of the documents, called the stock list, stores a list of the products currently stored in the shop, that isn’t available for sale. It also stores the quantity of that product in the stock room. Furthermore, the data the list was created is also stored at the top of documents.

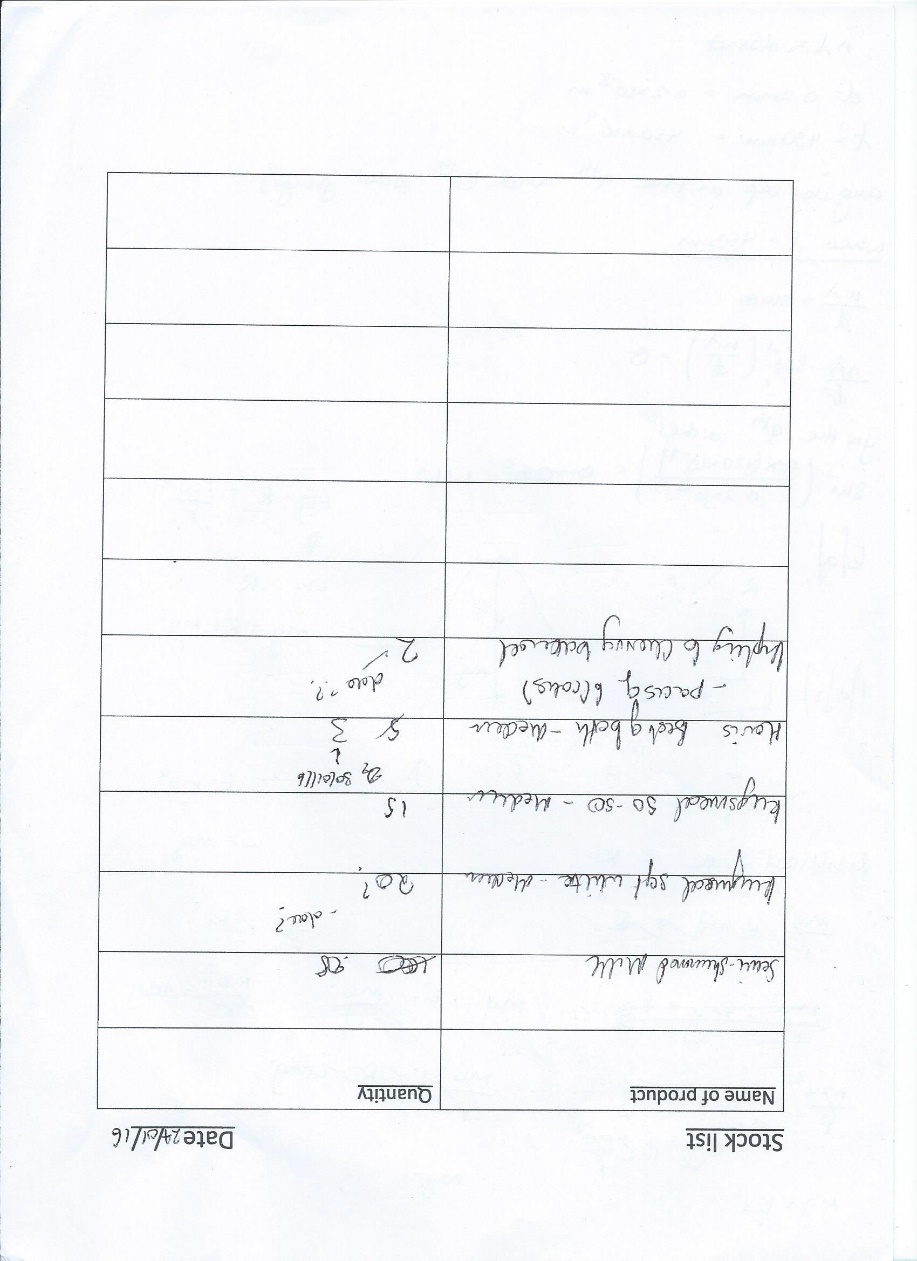


Figure 1

Shown above (Figure 1) is a scan of a stock list sheet used earlier on this year.

In addition to the stock list paper document, the shop also makes use of another paper document, called the for sale list. This document stores the list of products being taken out of storage and on to ‘the shelf’, making it available for purchase. The list stores information such as the name of the product, the quantity of that product and the expiry date of that product, should it have one. It also stores the date the list was created.

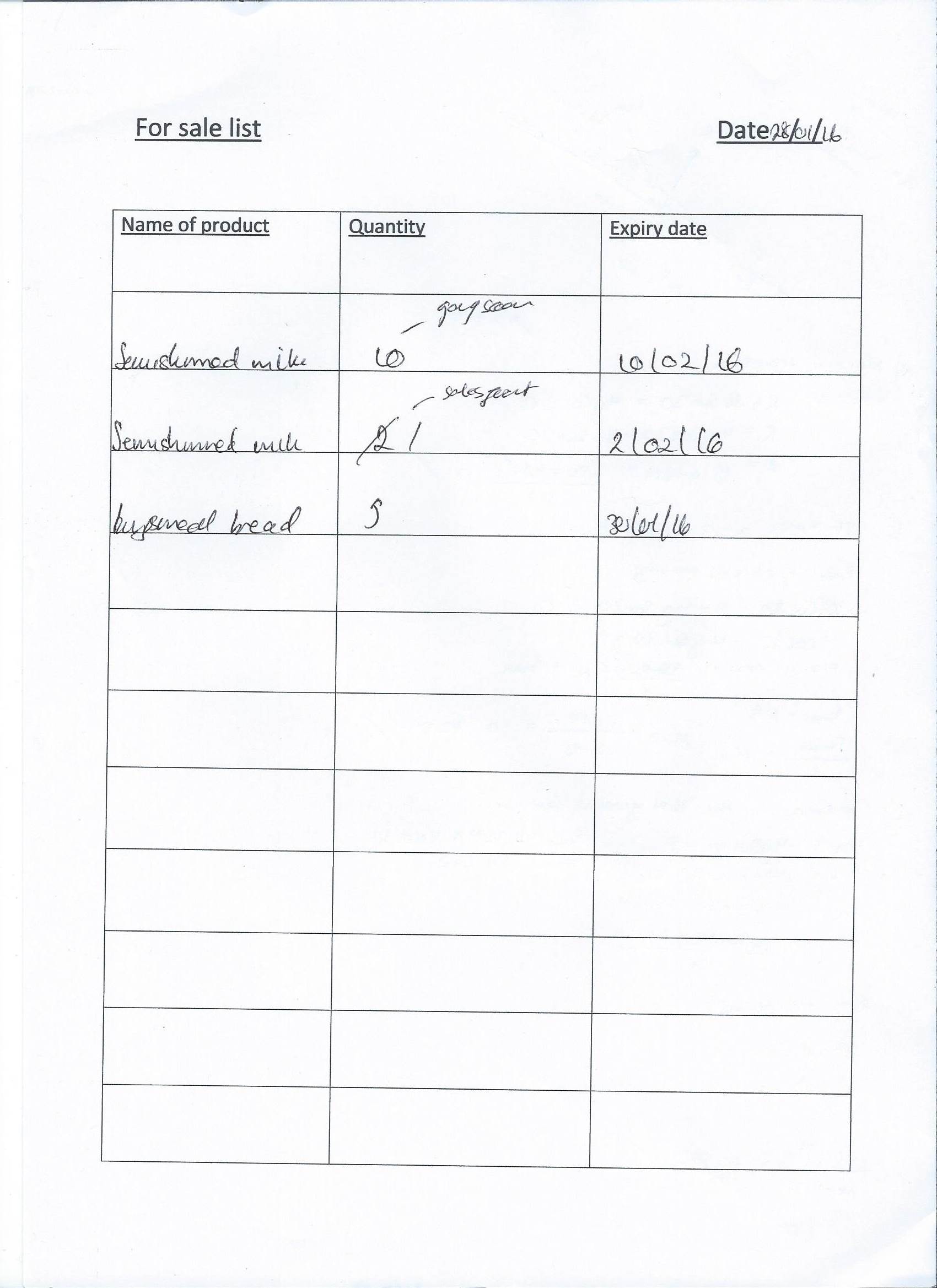


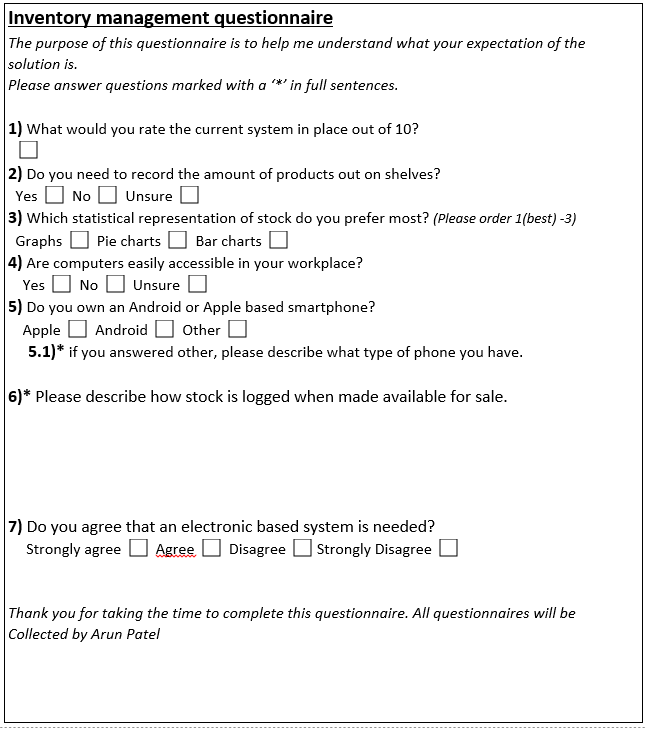
Figure 2

Shown above (figure 2) is a scan of an empty for sale list sheet.

## 4 Data collection

At this stage, I am going to get as much information from the client as I can regarding what the client wants from the solution. This will ensure that I produce a solution that does exactly what the client wants it to do. This will also allow for strong communication between myself and the client, meaning less time will be wasted in the future and the chances for mistakes to be made are reduced.

### 4.1 Questionnaire

The following questionnaire was given to my client and staff members to complete.

My client’s answers were as follows:

**1)** What would you rate the current system out of 10?

Answer: 4

**2)** Do you need to record the amount of products out on shelves?

Answer: Yes

**3)** Which statistical representation of stock do you prefer most?

Answer: Graphs (1), Bar charts (2) and Pie charts (3)

**4)** Are computers easily accessible in your workplace?

Answer: No

**5)** Do you own an Android or Apple based smartphone?

Answer: Android

**6)\*** Please describe how stock is logged when made available for sale.

Answer: When stock is put out on the shelf, it is initially taken off of the stock list. It is then

Added to a for sale list. If the product is an item that has an expiry date , that date is then recorded on to the list as well , so we know when stop selling a certain product. We also record the amount of that certain product taken out, for sales analysis purposes.

**7)** Do you agree that an electronic based system is needed?

Answer: Agree.

From the answers my client has given me for this questionnaire I can deduce that my solution should present statistics in the form of graphs and bar charts. I also understand that my client wants the new solution to be created.

I have also identified that my client does not have easy access to computers, this means I will have to find a way of allowing my client to be in use with the system whilst not a computer. One possible solution to this is to make use of a smartphone. My client has an android based smartphone, this means I could build an app to go alongside the main computer based system. This will make it easier to record the number of products out on the shelves.

I have also gained an understanding for how the stock is logged when made available for sale, this is an essential part of the system that would have to be accurately modelled in the final solution.

The questionnaire was also given to 3 other staff members at the business. The results of those questionnaires have been summarised below (excluding question 6). I processed my results using the charts feature in Microsoft Office Word (Microsoft, n.d.)

Question 1 -

Question 2 – All staff members ticked the yes box

Question 3 –

Question 4 –

Question 5 –

Question 7 –

### 4.2 Interview

#### 4.2.1 Face to face interview with client summary

I had arranged to have a face to face interview with my client at the workplace. I arranged for this interview by emailing my client and asking them if it was ok for me to conduct the interview and at what time and day would suit them best. The aim of the interview was to gain as much information as possible, mainly about the client themselves, such as their computer preferences, workload and computer skills. I also wanted to see, in person, exactly how some of the processes such as receiving new stock are actually conducted.

After agreeing on a time and date for the interview, I made my way to the business and conducted the interview. Initially, my client showed me the current system and how it works, he showed me the process of how new products are delivered to the store, added to the logs and stored in the stock room. After, he showed me the process of taking out stock from the stock room and making it available for sale.

The main issue with these processed seemed to be the amount of time it took to complete the tasks. The client seemed to get frustrated when trying to find items on the stock list as there was at least 100 items on the list.

After showing me a demonstration of the current processes the client undertakes, he then went on to give some ideas that could be added to the system. He suggested that the main system be able to search for products stored in the system. He also suggested that the new system maintain two separate lists, for products in the stock room and products for sale.

The client also raised an issue about the new solution, the fact that it wouldn’t be practical to use whilst performing everyday tasks. He then went on to explain that with the current system, he is able to walk around with the pen and list and make changes anywhere, whilst with a computerised solution, he would have to be at a computer which he cannot easily carry out with him.

#### 4.2.2 Face to face interview with client transcript

The following is a transcript of the interview I had with my client

*Me:* The first question I would like to ask, is what happens to new items when they have been delivered.

*Purnesh:* When we receive a delivery, the first thing that happens is that the person who delivered the items and myself count the quantity of items that have arrived, to ensure that we have received everything we paid for. After, I then sign a form to confirm receipt of the items. I then pass over receipts to other staff members who will then put away and store the items. This is done by writing the name of the item on the stock list, as well as its quantity.

*Me: Ok, so all items in the shop are recorded the stock list sheet?*

*Purnesh:* All items in the stock are logged on the stock list sheet but items out on shelves are logged on another sheet, the for sale list sheet. Items are transferred to this sheet when moving stock from storage to the shelves.

*Me:* Could you please describe the process of transferring stock from storage to the shelves

*Purnesh:* Yes, when an item on the shelf needs to be replenished or an item needs to be made available to customers, a staff member will firstly need to find the item in the stock list remove it from the list or amend its quantity , based on the amount of that item to be taken out. The item and quantity of that item is then added to the for sale list. If the item has an expiry data, it is added to the list. The actual item is then moved to the relevant location in the shop.

*Me:* Have there been any issues with this process?

*Purnesh:* There have been many issues such as staff members making mistakes, changing the details of the wrong item, not being able to tell which staff member has made changes and not being able to find certain items since the stock or for sale paper containing that item had been lost. The main issue with doing this, is that it is very time consuming and inefficient. It can sometimes take hours to work through hundreds of items at a time in order to find a particular item.

*Me:*  Ah ok, the new system will mainly be based around the use of a computer, what type of computer do you have?

*Purnesh:* I have a relatively modern desktop computer over near the counter area. It’s not very fast but good enough to get jobs done.

(My client then went to show me his desktop computer)

*Me:* Do you have much computer experience, how confident would you say you in getting to grips with new software?

*Purnesh:* I’ve been using computers for many years however, I would not consider myself ‘good’ with computers. I am able to use software such as the office word and excel but I’ve never really been exposed to new software. I think I would need a lot of help to get started with it.

*Me:* The new system can be made in a way that is user friendly to new users. Do you have any concerns about moving from a paper based system to a computerised system?

*Purnesh:* Yes, with a computerised system, I feel as if I would be stuck at the computer when using it and I would have to move around the shop and back to the computer multiple times. For example, at the end of the day the actual stock on the shelves are compared with the stock on the ‘for sale list’ and changes are made. If I had to do this with a computer, I would have to check the stock on the shelves and move back to the computer over and over again.

*Me:* Ok, that could potentially be resolved by making use of a smartphone. Do you have any suggestions as to what you would like to see in the new system?

*Purnesh:* I think the new system should be able to easily search for items, it shouldn’t take up to an hour to find an item in the list, like it currently does. I think maintaining two separate lists for items in the stock room and products for sale would be suitable.

*Me*: Ok, thank you for your ideas and your time today.

*End of interview.*

## 5 Diagrams

### 5.1 The current system

Here is a summary of the processes undertaken in the current system in the form of an Activity diagram.

#### 5.1.1 Receiving new stock

Y:\downloads\Adding stock flow.png

#### 5.1.2 Making stock available for sale

y:\downloads\MakeForSaleDiagram.png

### 5.2 Capabilities and Limitations of the current system

#### Capabilities

The current system is able to store a list of items currently held in inventory as well a list of items currently for sale.

It is able to remove items from both lists, change details about those items as well as allow staff members to read the data held about the items.

#### Limitations

Transferring items from the inventory list to the for sale list can take a long time and can lead to errors being made.

Changing the details of each item can only be done a finite times as it would be amended on a paper based system.

It can take staff members a lot of time to find and read the data about items. The system is not able to track who makes changes to the data being held in the system, for example, if a staff member adds a new item to the inventory list , there is no way of determining which staff member added that item.

It is not able to analyse trends about the data being held, it would have to be done manually by a user. It not able to predict which items will sell better and which won’t.

The data about the items is only accessible in one place. If a user is another location to the system, they will not be able to access the data, which will prevent them from finishing their work.

## 6 The proposed solution

### 6.1 Solution requirements

#### 6.1.1 Hardware requirements

* A computer with an internet connection
* This will allow the user to make use of the solution I have created.
* A smartphone
* Since there is not easy access to computers, a smartphone can be used instead when not at a computer to complete simpler tasks.
* A Keyboard
* This is a basic input device used to input characters into the computer. It is essential for both the users of the system and myself, to use the computer.
* A Mouse
* This is another basic input device used to allow the user to move a mouse pointer around the screen. This is necessary for the basic operation of the computer.

##### Required computer specification

The minimum hardware requirements that a computer needs to run this software is the minimum hardware requirements to be able to run Microsoft’s Windows 10 64-bit (Microsoft, 2015), however, for an optical experience, the user should have a computer whose specification are in line with the recommended hardware specification.

Minimum hardware specification

Processor – any processor with a clock speed of 1GHz or more

RAM – at least 2 GB

Hard disk space – at least 21GB

Graphics – Direct X 9 or later graphics card.

Display – a minimum of 800x600

Recommended hardware specification

Processor – any processor with a clock speed of 2GHz or more

RAM – at least 2 GB

Hard disk space – 30GB or more

Graphics – Direct X 9 or later graphics card.

Display – 1920x1080 or higher.

#### 6.1.2 Software requirements

* A desktop operating system – Windows 10/8/7 OS
* The solution will be created using the C# programming language and will therefor only run on a Microsoft Windows Operating System. This means that the users of the solution must have access to either a Windows 7 , 8 or 10
* A mobile operating system – Android based
* The results of the questionnaire revealed that the client has an android based phone, therefore it a software requirement to have an android based phone.
* An Integrated Development Environment (IDE) - Visual Studio / Eclipse
* In order to create the solution efficiently and effectively, I will need a programming IDE to help me write code.
* Eclipse will be used to create java based applications, the language used to create android apps.
* Photo editing package – Adobe Photoshop
* The solution may require graphics to be created, or to edit existing graphics to suit the solution.
* Microsoft Office suite
* The office suite is used to create documents, slide shows and spreadsheets, which all necessary for the development of this project.

#### 6.1.2 Solution requirements / success criteria

##### Functional requirements

* The solution must be able to input new stock items.
* This is essential for entering new items into the system when a delivery arrives.
* Entering the new details about an item into the system should cause the system to retain this information in the database connected to it.
* The solution must be able to take out stock items
* This allows the user to make stock available for purchase.
* Choosing an item , its quantity and making the request to make it available for sale should cause the system to move the item from the inventory list to the for sale list.
* The solution must be accessible from different computers
* This allows the user to access the database from different computers. This is needed as the client will be using different devices to make use of the solution.
* If a user uses the solution on a computer at the shop and then at a computer in their home, the solution should still be fully functional.
* The solution should be useable by inexperienced users
* The client is not very experienced with computers therefore the solution must be user friendly enough for inexperienced users.
* The user as well as all staff members should be able to use the solution without any additional help from anyone else and by only using help provided in the solution.
* The solution must be mobile
* The user isn’t going to be able to easily access a computer, therefore the solution must be useable whilst “on the move”.
* The user should be able to walk around the shop and be able to change the quantity of items in the for sale list based on the actual quantity on the shelf.
* It must have a facility to change the prices of products
* The prices of products often change, therefore the system must be able to edit the price of a certain product.
* The user should be able to enter the new price of an item and its new price should be updated on the items record in the database.
* It must be able to find popular products
* In order to continue selling products that are generating the most profit, the system must be able to identify these products and tell the user.
* If a product is moved from inventory to for sale and the amount of that item on the shelf everyday decrements frequently then the system must detect this and notify the user.
* It must be able to find unpopular products
* Products that are not selling well should be made to the attention of the user so that they can stop buying more of it.
* If a product is rarely moved from inventory to for sale and the amount of that item on the shelf everyday doesn’t change frequently then the system must detect this and notify the user.
* It must be able to predict when products will sell well
* Products will sell be better at certain times of the year, the system should learn this trend for certain products and notify the user.
* If a product such as ice cream sells more during the summer and worse during the winter, the system can detect this and notify the user.
* Data stored in the solution must be valid
* This will ensure that the data stored in the system is free of errors , reducing the chance of an output being incorrect as a result of incorrect processing
* Entering data into a validated field should present the user with an error.

##### Non-functional requirements

* The solution must be able to load in under 20 seconds
* This will ensure the user is not left waiting for the program to load when they initially open the program. This is dependent on the size of the database (the number of stock items stored), speed of the computer and speed of the internet connection.
* The system should load and be fully useable within 20 seconds.
* The solution must have a security login system
* This will ensure that unauthorised access does not occur. Since the solution will store data about the business, the solution must be secure enough to prevent anyone from accessing this data. Providing a login system will also allow the system to track who has been using the solution.
* When the system starts up, it should prompt the user for a login ID and password. It should only give users access to the system if the credentials they entered matches the details in the database.

##### User requirements

Due to the nature of the solution and the hardware available an existing manual process of physically counting the number of items out on the shelves at the end of the day must remain. A solution to this would be to use a handheld stock taking bar code terminal such as the “ERS-STOCKCOUNT-KIT1 ERS STOCKCOUNT Barcode Stock Taking Solution” (ERS, 2016) . The cost of such equipment is approximately £700 per unit, which is not always viable for a small business. However this process could be simplified with the addition of a mobile based solution.

### 6.2 Summary of the solution to be created

The solution I am going to create is an inventory management system for my clients shop. It will be capable of adding new items that has been delivered to the shop into the system, changing any details about that item and removing that item from the system. It will be capable of differentiating between items that are currently for sale and items that are in the stock room (not currently for sale).

Due to hardware limitations (see above), there will also be a smartphone based solution for the android operation system. This will be provide the basic feature of being able to log the quantity of products that are actually “on the shelf” at the end of the day. This will make it much easier to do this since the user can simply walk around the shop with their smartphone, logging the quantity of each product.

This leads to the main Unique Selling Point of the solution. The solution will make use of Artificial intelligence, Machine learning and statistical analysis to make predictions and analyse the sales performance of products. For example, Artificial intelligence can be used to make decisions on which products should be stocked more or less. If a new product does not sell well, a decision can be made by the solution to not order any more of that product. Similarly an example of were machine learning can be used in the solution is to predict the time of year a certain product will sell more. For example, the solution can learn that ice creams sell well during the summer and not as well in the winter. This means that the solution will understand that some products are seasonal. Statistical analysis will be used on the data collected from main solution and smart phone solution to come to these conclusions.

#### 6.2.1 Changes to the solution

Due to the programming languages and platform to be used in the final solution, I will no longer be able to use Google’s Firebase. This means the solution will no longer have a real time aspect to it, however, this does not mean changes will not be seen across instances of the solution open at the same time but there may be a small delay. Google’s Firebase does not have support for SQL, which is needed in order to create an effective solution.

Instead of using Google’s Firebase, I will be using Microsoft Azure SQL Database (Azure, 2016). Microsoft Azure SQL Database offers more support for C# developers and integrates well with the IDE I will be using, Visual Studio. In addition, it also has support for Java developers, the language I will be using to create the smartphone solution. Microsoft Azure SQL Database is also a pay as expand product, which means the business will only need to pay more for their services as the business grows. This will be due to the amount of inventory being held by the business expanding and so the amount of data being held in the database expanding meaning it requires more resources to hold the data leading to having to pay a greater fee to use the service. Despite this, initially there will be no cost to use their service.

## 7 Signatures

Confirmation of hardware requirements: Date of signature-

Confirmation of functional requirements: Date of signature-

Confirmation of software requirements: Date of signature-

Confirmation of non-functional requirements: Date of signature-

Confirmation user requirements: Date of signature-

# Design

## 1 System objectives

Although I have already defined solution requirements above, by defining system objectives I am able to expand on the solution requirements in order to deduce what type of interface the system should have based on the solution requirements. Each design created will follow the applicable functional requirements listed above. Some of the objectives outlined here can also be classed as primary objectives; required features that must be added to the system.

### 1.1 User interface

* Background colour will be plain white.
* It must use fields for data entry.
  + To allow the entry of inputs described in the system requirements
  + These fields must have a black border to show contrast against a white background.
* A navigation bar will be present on every page on the left hand side of the interface
* An exit and minimise button must be displayed on the top right hand side of the interface
* The interface should be of suitable dimensions to view on a screen with a resolution of 1920x1080 pixels.
  + This is not required of the mobile based application.

### 1.2 Input

* The user will be able to add new items into the system.
  + These changes will be seen across any device logged into the system
* The user will be able to make certain items available for sale.
  + The user will be able to search for items
  + The user will be able to add an expiry date to an item if needed.
* The user will be able to record who has made changes to the inventory.

### 1.3 Processing

* A database will be created on initial use.
* Items are added to this database in real-time
  + The database will be able to store the name of an item, the quantity, its expiry date (if applicable) and whether or not it’s for sale.
  + The user will be able to order the table (e.g. in alphabetical order)
* A query is ran on the database for the item being searched, the user can then make changes on the returned record.
  + Changes include making an item available for sale, adding an expiry date, removing the item and adding more of the same item (CRUD).
* The items moved to for sale the most during the week are recorded summarised.

### 1.4 Output

* A real-time view of the database of items presented on the interface.
  + This will include, item name, quantity, and whether or not it’s for sale.
  + This will also be presented in a mixed table – containing all items , regardless of whether or not it’s for sale , a for sales table – containing all items that are currently for sale and an inventory table – containing all items that are currently in storage , not for sale.
* A weekly performance report. Products that are taken out of storage and made for sale are considered successful products (as they are in demand). The most successful products can be summarised on the report.
  + Graphical analysis will also be utilised here.

### 1.5 Internal Objectives

These objectives are objectives that I need to achieve in order to develop the solution.

* Learn SQL
  + Structured Query Language (SQL), is the language that is used to manipulate and access databases. It is therefore essential that I am comfortable with the language in order to create an effective solution
* Learn how to setup and use Microsoft Azure Database.
  + Microsoft Azure Database is the tool I will be using to store data being stored in the system. Without it, the solution would not be able to store any data.
* Learn how to use Android Studio
  + This is the IDE that will be used to create the smartphone application.

## 2 Scheduling

In order to produce an effective solution, I have made the following Gantt chart to illustrate my plan of how I will develop the solution.

### 2.1 Design plan

### 2.2 Development plan

## 3 New System

### 3.1 Use-Case diagram of new system features

The purpose of this use case diagram is to show how each user can utilise the system.

E:\OCR A Level Computer Science project\Design\Diagrams\ManagerUseCase.png

#### 3.1.1 Use-Case descriptions

Add items to stock list

*Summary*: Add new items to the system that are being stored in inventory.

*Precondition*: New items have been delivered.

*Actors*: Manager, Senior Staff, Staff

*Description*:

***Best case***

1. User enters the name and quantity of the item(s) being moved to inventory.
2. The new items are added into the system

***Worst case***

1a- The user enters the wrong item name or quantity

The system takes the user back to the data entry section of the system

Remove items to stock list

*Summary*: Remove items from inventory

*Precondition*: There are items in inventory to remove

*Actors*: Manager, Senior Staff, Staff

*Description:*

***Best case***

1. User enters the name and quantity of the item(s) being removed
2. The items are then deleted inventory table

***Worst case***

1a- The user enters the wrong item name or quantity

The system takes the user back to the data entry section of the system

Update items in stock list

*Summary*: Allows the user to amend the data held about items

*Precondition*: There are items stored in inventory

*Actors*: Manager, Senior Staff

*Description*:

***Best case***

1. User enters the name of the item to be amended.
2. The system searches and returns items related to the searched for item
3. The user selects the item the wish to amend
4. The user makes the changes on the item details
5. The system updates the data on that item

***Worst case***

2a- The system returns items but none are what the user wanted

The system tells the user to make their search more precise

Track who has made changes to the stock list

*Summary*: Allows the manager to track who has made any changes to the stock list table (or any other table)

*Precondition*: Senior staff and staff members have valid accounts with the system.

*Actors*: Manager

*Description*:

***Best case***

1. The user requests to see who has made changes to the inventory tables
2. The system searches for and returns a log of changes
3. The user can then view the log.

***Worst case***

2a- The log contains no data

The system tells the user that no changes have been made by staff members

As shown on the Use Case diagram above, the manager, who is my client - Purnesh Patel, has access to everything the solution is able do. Senior Staff members have access to everything apart from tracking and reports. Staff members, on the other hand, only have access to the adding and removing items feature , as well as being able access the for sale list , allowing them to make items available for sale, as well as changing or adding sell by dates on items stored in the for sale list.

The secondary solution, the smart phone application has a main purpose of item replenishment at the end of the day.

### 3.2 Input Output diagrams

The following Data Flow Diagram (DFD) is an extension of the system objectives outlined in the previous section. It will represent the different processes and data involved in the system objectives.



### 3.3 Inventory Database ERD

All of the data stored in the solution be held in a real time database. It will contain more than one table, making it a relational database. The following Entity Relationship diagram (ERD) summarises the structure of the database to be used with the solution.



### 3.4 Data Dictionary

A data dictionary defines the structure of the database itself. It records what data is stored, the name, description and characteristics of each data element.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field Name** | **Field Size** | **Data Type** | **Data Format** | **Description** | **Example** |
| User ID | 10 | String |  | User ID for each staff member , used to log in. | PPatel |
| Password | 10 | String |  | Password for authenticating logins. | password |
| Name | 20 | String |  | Used to store the name of the person corresponding to the User ID. | Purnesh |
| Job Title | 10 | String |  | This is used to identify the privileges of each user. | Manager |
| Item ID | 7 | String | LLLLNNN | Primary key of items stored in the database. | ITEM001 |
| Item Name | 20 | String |  | Gives the Item ID an actual product name. | Kingsmill bread |
| Quantity | 3 | Integer |  | This value is the number of an item in the table | 5 |
| Price | 8 | Currency | £NNNN.NN | The selling price of a particular item. | £3.20 |
| Category | 10 | String |  | The category a particular item falls under. | Food items |
| For Sale ID | 8 | String | LLLLLNNN | The primary key for items that are for sale. | FSALE001 |
| Sell by date | 10 | Date | DD/MM/YYYY | The date in which an item that is for sale will no longer be able to be kept for sale. | 01/01/2017 |
| Performance ID | 5 | String | LLLLNNN | The primary key for items who’s performance are being tracked | PMCE001 |
| All time quantity of item made available for sale | 3 | Integer |  | This is the total number of times the item has been made for sale since the first use of the solution | 5 |
| Quantity of item made available for sale past week | 3 | Integer |  | This is the total number of times the item has been made for sale for the current week. | 2 |
| Items |  | Multi-dimensional string array | Dimension 1 :  LLLLLLLLLL…L  (L characters)  Dimension 2:  NNNN | A multi-dimensional array used to hold the contents of items retrieved from the database. The index 0 of the first dimension represents the name of the item. The second dimension represents the quantity of that item. | Index 0, 0 could have the value Kings meal white bread. Index 0, 1 could have the value 10. |
| Selection |  | String array | LLLLLLLLLLLL | A string array used to hold a list of items that the user has selected from either the inventory list or the for sale list. | The value at index 0 could be Kings meal white bread. The value at index 1 could be Semi skimmed milk. |
| Best-  Performed-  Items | Bounds: maximum 5. | String array | LLLLLLLLLLLL | A string array used to hold the best performed items for that week (or any other given range). | The value at index 0 could be Kings meal white bread. The value at index 1 could be Semi skimmed milk. |
| Worst-  Performed-  Items | Bounds: maximum 5. | String array | LLLLLLLLLLLL | A string array used to hold the worst performed items for that week (or any other given range). | The value at index 0 could be Kings meal white bread. The value at index 1 could be Semi skimmed milk. |
| Authenticate | 1 | Boolean | True/False | A Boolean flag used in the logging in process. The flag is false if the inputted credentials have not yet been authenticated. The flag is changed to true once a successful, authenticated, login has been made. | An incorrect password has been entered, so the value is false.  The authentication was successful so the value is set to True. |

### 3.5 Database normalisation

Database normalisation is the optimisation of a database to make it organised in such a way that there is no data duplication , data is consistent throughout the database , the structure of each table is flexible enough to enter as many or as few items needed and it should enable a user to make complex queries relating data from different tables.

#### 3.5.1 Zero Normal Form (0NF)

This describes the state database is in, in its normalised form. The database is most likely to run into errors and is least efficient and effective at this point.

#### 3.5.2 First Normal Form (1NF)

In this form, all the tables in the database contain no repeating attributes or groups of attributes. Each data item cannot be broken down any further, each row must contain a primary key and field must have a unique name. An example of how the database used in solution could be normalised to the First Normal Form is by deleting repeated data is making sure each row contains a primary key, for example, in the items table, giving each item an item ID will give each row a primary key, the database is now in the First Normal Form (provided it meets the other rules).

#### 3.5.3 Second Normal Form (2NF)

For a database to be in the Second Normal Form, It must first be in the First Normal Form. It also must ensure that non-key attributes depend on every part of the primary key. This means that any database in the First Normal Form with simple primary keys (not compound primary keys) will automatically be in the Second Normal Form. Normalising to this form ensures no redundant data is being stored. An example of normalising to this form in the solutions database would be changing a table with primary keys Item ID and Performance ID. It also contains the attribute Price. This attribute is relevant to the Performance ID key but not the Item ID key, therefore preventing the database from being in the Second Normal Form. In order to normalise this database, another table must be created, which will hold the price of the items. It could be given the primary key Price ID.

#### 3.5.4 Third Normal Form (3NF)

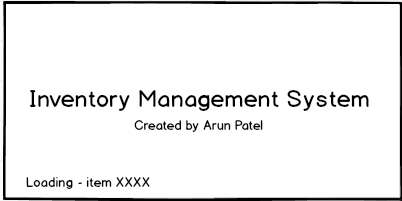
To normalise to the Third Normal Form, the database must already be in the Second Normal Form. In this form, there are no non-key attributes that depend on another non-key attribute. Similarly to 2NF, 3NF tries to remove another source of redundant data. It does this by making sure the value of one attribute cannot be worked out looking at another attribute. For example in the solutions database, it may contain the attributes quantity, price and revenue, however, since the price and revenue are given , the quantity can of the items could be worked out , by dividing revenue by the items price. This could database could be normalised by creating an additional table, Quantities, and moving the quantity of each item to that table.

## 4 User Interface

In this section I will be designing the user interface for the different sections of the system.

### 4.1 Splash screen design

Figure



400px

200px

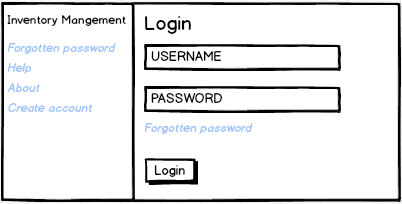
This shows the current item being loaded from the database

Figure 3 is a design of the systems splash screen, the first window to be displayed. It will remain on the screen until all of the items in the database has been loaded. This could take varying amounts of time based on internet connection speed.

### 4.2 Login screen design

400px

200px



This button allows the user to login to the system, provided they have entered valid login credentials.

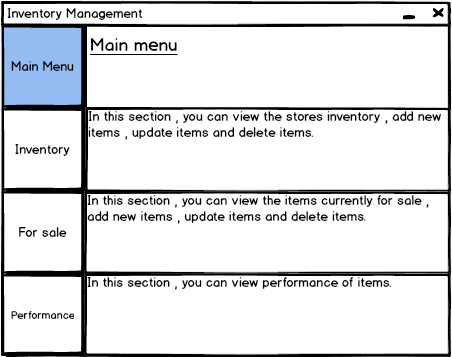
Figure

These are a clickable labels that allow the user to execute different tasks

Figure 4 shows the design of the second screen to be displayed to the user. This screen will be displayed to the user without rendering a new windows but instead will overwrite the contents of

### 4.3 Main menu

600px



The large sidebar buttons are the main navigation system of the system. Currently, the Main menu button is illuminated since the user is on the main menu page.

Figure

800px

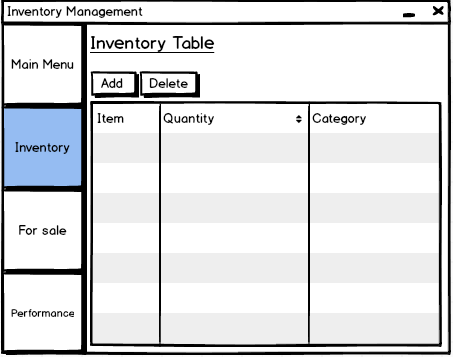
These descriptions are short summaries of what each page allows the user to do. Clicking the large buttons associated to these summaries will take the user to that page

Figure 5 shows a design of the main menu of the solution. Shown on the left hand side of the interface is the navigation system that will be present on every page of the solution. Clicking the large buttons will result in the user being taken to the relevant page of the interface.

### 4.4 Inventory page

600px

800px



This table is a direct representation of the inventory table in the database and should display live data.

Figure

These are buttons allowing the user add new items or remove a selected item.

The inventory button is currently illuminated as the user is on the inventory page

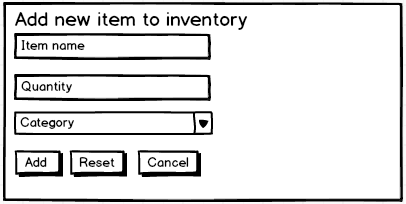
The user is able to select rows of items in the table to delete or change the data about an item.

Figure 6 shows a design of the inventory screen. On this screen, the user is able to view a list of items stored in inventory, loaded and updated in real time from the database. The user is able to select items (or a single item) from the list and delete the item. The user is also able to edit details of any items by simply clicking on an item in the list and changing its field’s details.

Furthermore, the user is able to click on the add button, allowing the user to add a new item into the inventory. Clicking the add button causes the following window to be opened (fig 4.4.1).

200px

400px



Figure

The reset button causes all the fields to become empty. The add button completes the add action and the cancel button closes the window

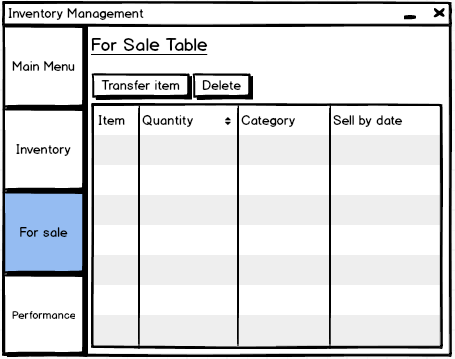
When this screen has been presented to the user (figure 7) is able to enter the details of a new item. These details include the name of the item, the amount of that item (quantity) and its category (an example of a category is food). The user then clicks on the add button, causing the new item to be added to the database and to be shown on the inventory table. Clicking the reset button causes the input fields to be reset to their default state. The cancel button causes the window to be closed and no changes will be made.

### 4.5 For sale page

This table is a direct representation of ‘the for sale’ table in the database and should display live data.

600px

800px



Figure

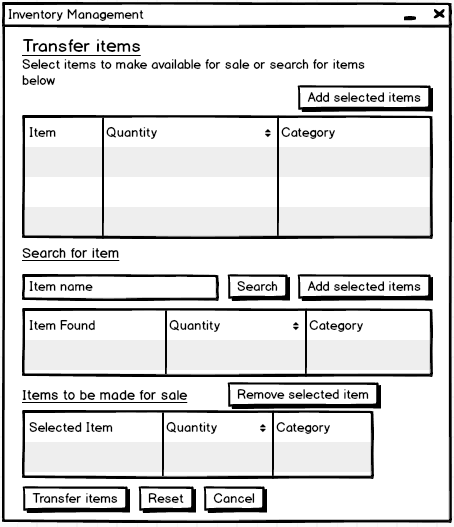
These are buttons allowing the user to transfer items or remove a selected item.

The For sale button is currently illuminated as the user is on the ‘for sale page’

The user is able to select rows of items in the table to delete or change the data about an item.

Figure 8 shows the design of the for sale page. Here, the user can view a list of all the items currently available for sale, loaded and updated in real time from the database. As with the inventory table, the user is able to change details of any fields by simply clicking on the fields and changing their values. If the user wishes to remove a selected item from the table, they can simply select the item(s) from the list and click delete, causing them to be removed from the database and the shown list.

Transferring items from inventory to the ‘for sale’ table requires the user clicking transfer item button. They will then be presented with the following window (figure 9).



600px

950px

Figure

All items in the inventory table are shown in this condensed (scrollable) table.

This button adds the selected item to the list of items to transfer (see below)

The names of items similar to those inputted in the item name are shown in this table.

The list of items selected, to transfer to ‘for sale’ are shown in this table. The user can remove items by selecting them and click on the remove selected item button

Transfer items completes the action on the items selected

On this screen (figure 9) the user must select the items from the inventory list to transfer to the for sale list. There are two ways the user can do this, the first being manually selecting the items from a copy of the inventory list and determining the quantity of that to add. Once the items have been found, the user can click the “add selected items” button, causing the selected items to show in the table at the bottom of the page, labelled “Items to be made for sale “.

The second method involves searching for the item to be added. The user can input the name of the item in the Item name field and click search, causing a query to be ran. The results of that query are shown in the table underneath the field. The user can then select the item needed, specifying the quantity to add. They can click on the “Add selected items “button, causing the items to be added to the list at the bottom of the page.

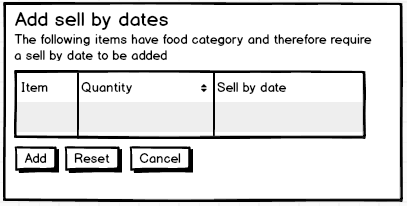
The user is also able to remove items from the list of items to be made for sale table by selecting records in the table and clicking the “Remove selected item” button.

The Reset button at the bottom of the page causes all the fields and tables on the screen to be reset back their original state, allowing the user to start over again. The Cancel button causes the window to be closed resulting in no changes being made.

Once the user is ready to transfer the items, they can click on the Transfer items button. However, if some of the items are in the food category (or any that requires a sell by date) another window is presented to the user (fig 4.5.2). If not, then the items are added to the list and database.

200px

400px



Figure

The user can add sell by dates to a batch of items of single items.

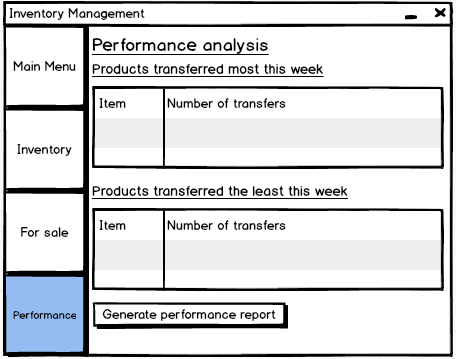
On this page, the user is prompted to add sell by dates to the required items trying to be transferred. The affected items are shown in the table on the window. The user selects the item in the list and adds the sell by date into the relevant field.

The Reset button reverts the page back to its original state and the cancel button takes the user back to the previous page (figure 10). Clicking the Add button results in the items (along with the items with sell by dates) being added to the list and database.

### 4.6 Performance page

600px

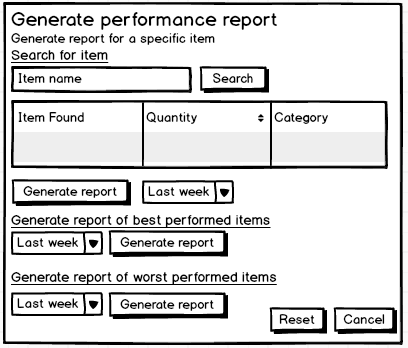
800px



Figure

The Performance button is currently illuminated as the user is on the ‘Performance page’

Figure 11 shows the design of the performance page. On this page, the best performed and worst performed products of the current week are shown on the page. The user can also generate reports by clicking on the “Generate performance report button”. The user is then presented with the following window (figure 12).



600x

750px

Figure

The names of items similar to those inputted in the item name are shown in this table.

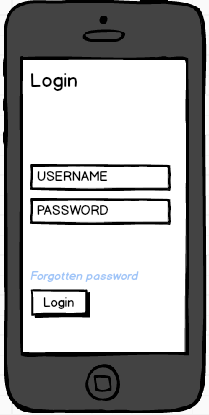
These buttons trigger a report to be generated based upon the inputs the user has given

On this window (figure 12), the user is able to generate 3 types of report; a performance report for a specific item, a performance report the best performed items and the performance report for the worst performed products. In all of these reports, the user can specify the range of data to use , for example , from the last week , month , year or from a specific range ( e.g. 10/12/16 – 1/12/16 ).

The user can search for a specific item by typing in the name of the item to find in the item name field and clicking search. The results of the search are shown in the table below the field. The user can then desired from the table and then select a range for the data to be used. Once they have done this they can click generate report.

The best and worst performed reports can be generated in a similar way. The user simply specifies the range of data to use and clicks generate report. The reset button causes the page to revert back to its original state, the cancel button causes the window to be closed, no changes or reports will be made.

### 4.7 Smart Phone login screen design



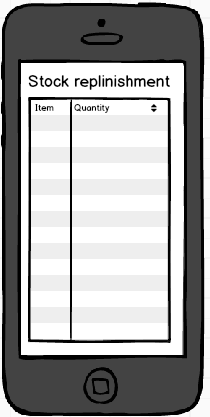
Similar to the desktop solution, the system prompts the user for a username and password.

Figure

This button allows the user to login to the system, provided they have entered valid login credentials.

Figure 13 shows a design of the smartphone login screen.

### 4.8 Smart Phone stock replenishment screen



Figure

This a list of items held in the for sale list.

The user can select an item and amend its data whilst walking around the shop.

Figure 14 shows the stock replenishment screen for the smartphone solution.

## 5 Data capture

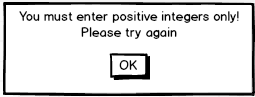
Data is inputted into fields located on the main user interface. The user is also able to input data whilst selecting specific records on the tables displayed in some sections of the interface.

### 5.1 Validation

Validation is an automatic computer check to ensure that the data entered is sensible and reasonable. However, it does not check the accuracy of data. Validation will be used wherever possible in the solution to ensure that data is as correct as possible.

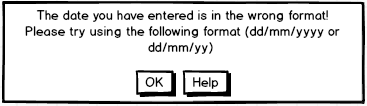
#### 5.1.1 Datatype checks

A Datatype involves checking the type of data that a user has inputted. For example, if the user inputs the following, “534”, the datatype check would identify the datatype of the input as integers. This type of check can be used to validate the input of the user when entering the quantity of an item to be added to the inventory database. By restricting the datatype to be of only positive integers I can prevent the user from entering incorrect data. For example, with the validation rule setup on the field, if the user tries to enter an input such as “w34”, the user will be presented with the following error. A suitable input would be “10”.



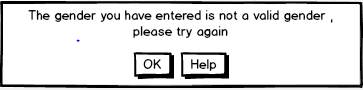
#### 5.1.2 Format checks

This validation check involves checking to see if the data is in the correct format. This can be used when the user is inputting the sell by date of an item when transferring an item. For example a valid format may be “dd/mm/yyyy” or “dd/mm/yy”, were d m and y are positive integers. A suitable input would be “01/01/2015” or “01/01/15”. However, if the user entered the following input “se/02/x4”, the validation check would detect an incorrect input and the following error message could be presented to the user.



#### 5.1.3 Table Lookup checks

A table lookup check takes the entered data item and compares it to a list of valid entries that are stored in a database table. If the entry is in the list of valid entries then it is allowed. If not, then the user is presented with an error. For example, if the user is asked to their gender when creating a staff account with the system, the solution can detect if an error was made as there are only two genders stored in the database, Male or Female. Any other inputs would not match the data stored in the table and so would cause the user to be presented with an error. The following error message could be presented the user if the validation check catches an error.



### 5.2 CRUD permissions

The solution will have permissions for CRUD (Create, read, update and delete) operations. The following user groups will be created, which will determine the permissions each user will have. Users will be placed into the group depending on their role in the business.

A summary of user groups are shown below in order of most power to least. User groups share the permissions of user groups below them.

#### 5.2.1 Admin group

This user group will have full CRUD access on both tables in the database as well as direct access to the raw database. No users of the system should be in this user group, its purpose is for debugging only.

#### 5.2.2 Manager group

This user group will have full CRUD access on both tables in the database but will not have direct access to the raw database. They also have access to reports and user tracking. This user group is for the main client of the solution – Purnesh Patel.

#### 5.2.3 Senior Staff group

This user group will have Create, Read and update access to both tables only.

#### 5.2.4 Staff group

This group will have CRUD access to the inventory (stock) list table but only have Read access to the for sale table.

The permissions are explained in further detail in the following table

|  |  |
| --- | --- |
| **Permission group** | **Permissions** |
| Admin | Full access to any features of the solution. It is only to be used for debugging purposes. It shouldn’t be used by the end users, they shouldn’t have access to this user group either. End users can use this permission group at their own risk |
| Manager group | This user group will have the most functionality. They will have access to all features of the solution intended for the end user. Whilst they won’t have direct access to the actual database, storing the data, they will have access to the database through the solution.  As well as this, they will have access to administrative tools such as tracking who has made changes to the inventory list or for sale list. The actual tracking log will be available to the user in this permissions group.  They will also have access to the item analysis section of the solution. Here they can view the performance of products.  They will not have access to the developer console. |
| Senior Staff group | This permission group offers most functionality of the solution but misses out some features reserved only for the manager group.  Users in this permission group will not be able to update items in the inventory table and will not be able to track who has made changes to any of the tables.  In regards to the for sale table, users in this group will be able to perform CRUD operations. In addition, they will be able to move items from inventory to the for sale table. |
| Staff group | This permission group is the most basic group the solution has to offer and offers limited functionality. It Is aimed towards staff members who are new to the shop or staff members who have little experience with computers. They will be able to add or remove items from inventory table and for sale table. |

## 6 Algorithms

The purpose of designing algorithms is to make it easier for the programmer to begin programming the system.

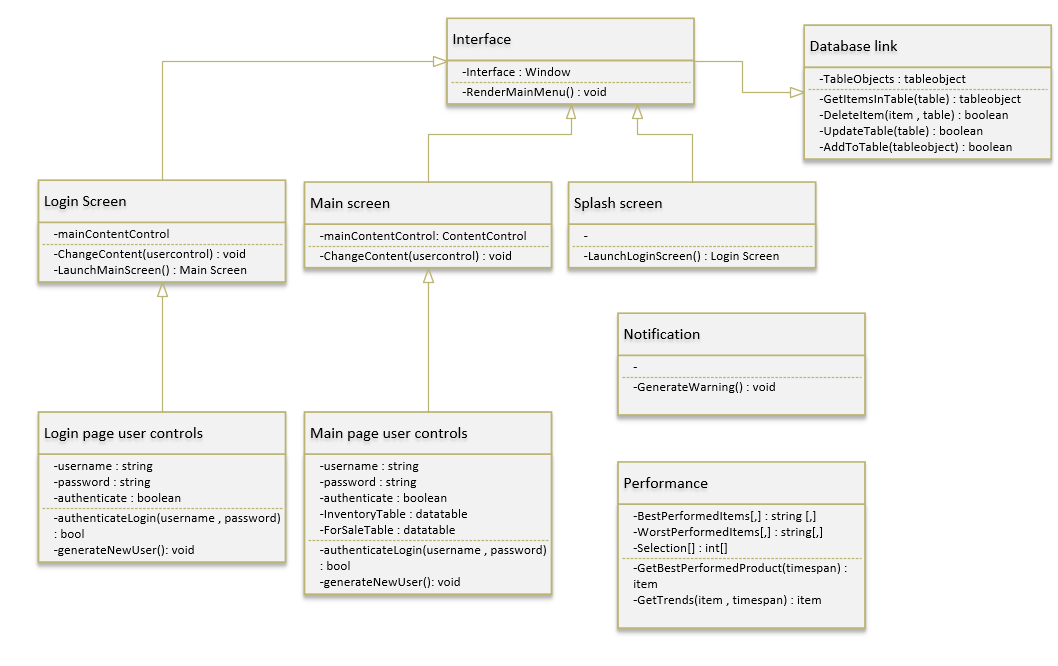
### 6.1 Structure diagram

I have created the following structure chart to show all of the sections of the system.



### 6.2 Class diagram

The class diagram below shows the classes, their subroutines and variables used in the system

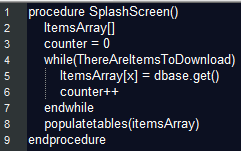


### 6.3 Algorithm design

In order to design algorithms for the different parts of the system, I will be making use of the structure diagram created above (6.1). The algorithms will be represented in pseudo code or flowcharts.

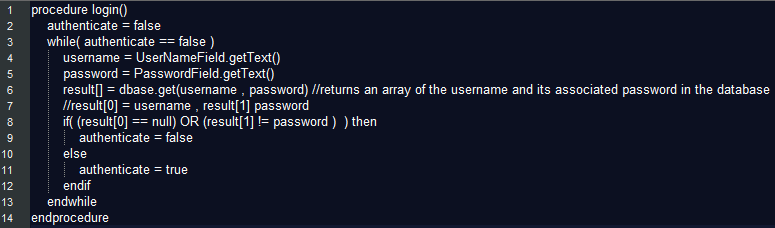
#### Splash Screen Loading

The following algorithm in pseudo code represents steps involved in initialising the system whilst the splash screen is open.



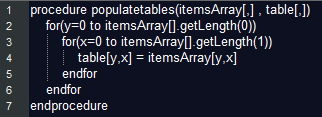
#### Login authentication

This algorithm for user logins will be used for both the computer solution and the smartphone solution.



#### Populate tables

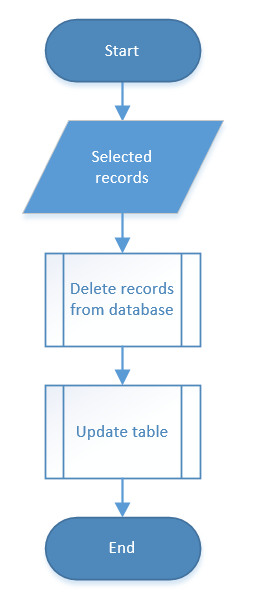
This algorithm is called when the splash screen has loaded the “itemsArray” array.



#### Adding new items to inventory

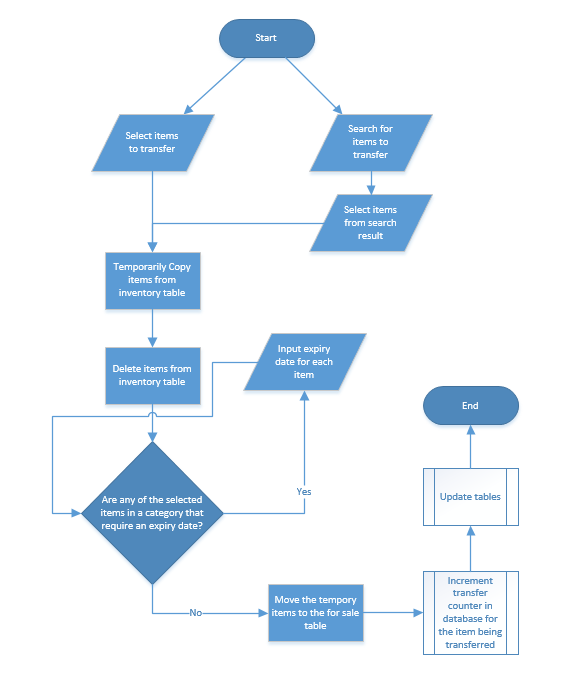
This algorithm assumes the system is fully loaded and describes the steps involved in adding new items to the inventory.

#### Removing items from inventory



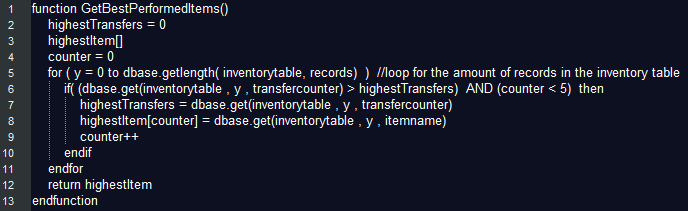
#### Transfer items

The user is able to select items in two different ways, both producing the same result

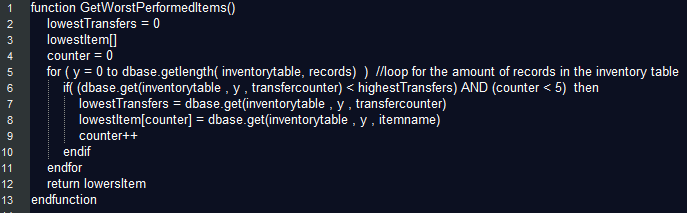


#### Get best performed items

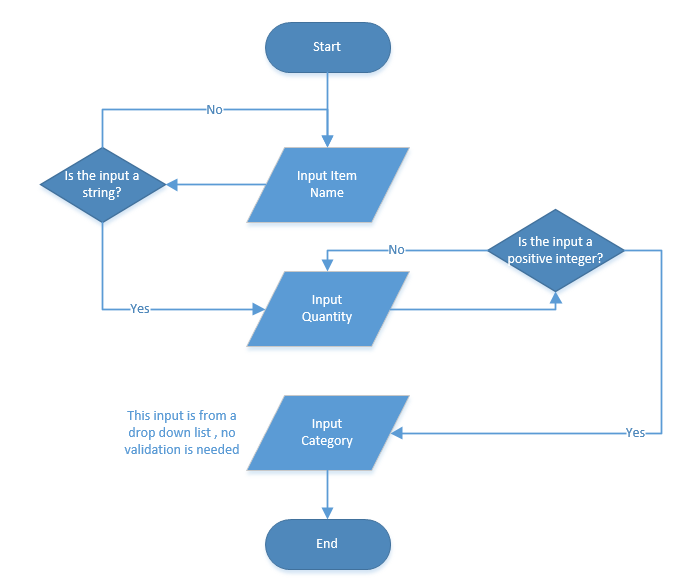
This algorithm finds the 5 best performed items. The performance of an item is determined by the amount of times an item is taken out of inventory and made available for sale. The process of logging transfers is shown in the Transfer items algorithm.



#### Get worst performed items



#### Validating input when adding new items to inventory



### 6.4 Database Queries

In this section I all will plan out the queries that will be ran on the systems database during operation.

The following list contains the types of queries the system will be able to run on the database.

* Select Query
  + This type of query is used to retrieve data from one or more tables and display the results
* Action Query
  + The database undergoes a specific action depending on what was specified in the query itself. This can include such things as creating new tables, deleting rows from existing ones and updating records or creating entirely new ones.
* Parameter Query
  + This type of query allows the same query to re used but with different criteria (a different parameter).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Query Name** | **Description** | **Query Type** | **Input fields** | **Output fields** |
| Search for an item in the inventory table. | This query will search for an inputted item stored in the inventory table. | Select query. | The name of the item to search for | The closest matching records in the database to the input field. |
| Search for an items quantity in the inventory table. | This query will search for an item in the inventory table and return its quantity. | Select query. | The name of the item to search for. | The closest matching records in the database but only including the item name and quantity. |
| Search for an item in the for sale table. | This query will search for an inputted item stored in the for sale table. | Select query. | The name of the item to search for. | The closest matching records in the database to the input field. |
| Search for an items quantity in the for sale table. | This query will search for an item in the ‘for sale table’ and return its quantity. | Select query. | The name of the item to search for. | The closest matching records in the database but only including the item name and quantity. |
| General item search | Searches all tables in the database to find an inputted item. | Parameter query. | Item name ,  Table (this can be all tables). | The closest matching records in the database to the input field. |
| User Authenticate | Searches the database for a username and its password. | Select query. | Username and password | Returns the record of the username if the password and username match the input exactly. |
| New user | Searches the database to check if the username already exists | Action query. | Username  -Account details | If there isn’t an existing record with the same username then create a new record with the inputted details. |

## 7 Test Strategy

The purpose of a test plan is to ensure that the client requirements are met, also, to ensure that there is valid inputs and accurate outputs. I have created a test plan to include the purpose of a test the test data and the expected result of that test. I will be testing the system myself, to ensure the functions of the solution works. Another way of testing, is to get feedback from the user on the system. I will show the result of the test (evidence) to show if it was successful or not. Should a test be unsuccessful, I will have to try and solve the issue and test it again to see if it’s working or not.

I will be carrying out functional testing in the testing phase. This will involve testing the functionality of the program as it is developed using normal, boundary and erroneous data sets. This will ensure that the solution works for whatever input it is given. Invalid inputs should be detected and brought to the attention of the user.

#### Data sets

Normal data is data within the range and data type you would normally expect. For example entering the quantity of an item should be a positive integer such as 5.

Boundary data is data at the ends of the expected range. In terms of the quantity of an item input, the boundary data set could be 1 to 1000. Although there is theoretically no limit on the upper bound of this data set, 1000 is likely the maximum quantity for a specific item in the business I am developing for.

Erroneous data is data that is either outside the expected range or of the wrong data type. For the quantity input, an erroneous data set could be “a” or -1. Here, “a” is of the wrong data type, an integer is required and -1 is outside the expected range, it is impossible to have negative one of something or 0 of something.

#### Dry-running

For some tests, it may be suitable to trial a dry run of the program. This can be helpful in locating a logic error in a program. In a dry run, I can follow through the logic of a particular part of the program in the same sequence the computer does with the aid of a trace table. In a trace table, I can note down when each variable changes and what its value is.

#### Black box testing

This will be main testing strategy. It involves carrying out tests independently of the code. It involves creating test data, based on the solutions specification that covers all the inputs, outputs and program functions.

#### White box testing

In white box testing I will be considering the solutions code, focusing on its logic. It focuses more the programs structure rather than its functionality. Tests are created which are designed to test each possible path at least once.

#### Alpha testing

This testing is carried out by the developer’s in house testing team. In my case I will be handing over my solution to family members who are experienced with computers to test the functionality of the system. The user may discover errors and omissions in the systems requirements definitions. Due to the scale of my solution, I will not be carrying out Beta testing,

#### Acceptance testing

This testing is used to determine whether or not the solution has met the requirement specifications. This testing will be carried out at the end of the development stage prior to release. Depending on whether my client will have time to do this, I plan to go through every solution requirement with my client and demonstrate them in the solution I have created. However, this can take a very long time and so my client may not want to do this.

### 7.1 Test plan

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test No.** | **Purpose of test and Solution Requirement** | **Data set** | **Action** | **Expected result** |
| 1 | Demonstrate the requirement :  The solution must be able to input new stock items. | *Normal*  Item name – Kings meal White Bread  Quantity – 10  Category – Food  *Boundary*  Item name – Kings meal White Bread  Quantity – 1  Category – Food  *Erroneous*  Item name – Kings meal White Bread  Quantity – A  Category – wef23 | On the inventory page I will click the add button. On the add window I will input the test data into the relevant fields. I will then click add. | The new item, with its details should be present on the inventory table on the UI and as an actual record in the actual database.  The same result should be given for the boundary data set.  An error message should be displayed when the erroneous data set is entered into the fields as it is an invalid input. |
| 2 | Demonstrate the requirement :  The solution be able to take out stock items | *Normal*  Item name – Kings meal White Bread  Quantity – 10  Expiry date – 20/11/16  *Boundary*  Item name – Kings meal White Bread  Quantity – 1  Expiry date – 1/1/0  *Erroneous*  Item name – Kings meal White Bread  Quantity – (-10)  Expiry date – FF/11/16 | I will firstly click on the transfer item button. I will search for the item, by typing in the item name in the item name field. I will then select the item and select a quantity from the search results and click transfer items. When prompted to, I will add the expiry dates to the item. | The transferred item should now show on the for sale list on the UI and on the for sale table in the database. The item should no longer show up in the inventory list on the UI and on the database, since all the items have been removed. This should be the case for the normal and boundary data sets. However for the erroneous data set, an error message should be shown. The quantity cannot be negative and the expiry date should not contain letters in. |
| 3 | Demonstrate the requirement:  The solution must be accessible from different computers | *Normal*  Computer in location A – internet connection  Computer in location B – internet connection  *Boundary*  Computer in location A – internet connection  Computer in location B – very slow internet connection  *Erroneous*  Computer in location A – internet connection  Computer in location B – no internet | In location A I will be add item on the inventory page and close the solution. In location B I will open the solution to check if the new item is present. | The new item added should be present in location B , with exactly the same details as the input in location A. This should be the outcome for both the normal and boundary scenarios. The boundary scenario should result in the computer at location B taking longer to load the items. In the erroneous scenario, the computer at location B shouldn’t be able to present any data due to lack of internet connection. If any data is displayed it should outdated data and shouldn’t include anything to do with the input at location A. |
| 4 | Demonstrate the requirement:  The solution should be useable by inexperienced users | Input new item into inventory. Move the item to for sale. Update its expiry date. Lookup performance information about that particular items history. | I will invite staff members at the shop who feel they are inexperienced with computers to use the solution. If they are able to complete the given task, then the solution can be labelled as useable for inexperienced users. | The staff member(s) should be able to perform the given task. |

## 8 Signatures

Confirmation of design stage: Date of signature-

# Development

At this point, I will now begin developing the solution. In this section of my report I will document my progress in creating the solution. I will be developing in a modular approach. These modules include, an interface module, a database module, database connectivity module (including Create, Read, Update and Delete) and an AI and machine learning module. To start with I will begin with the interface module.

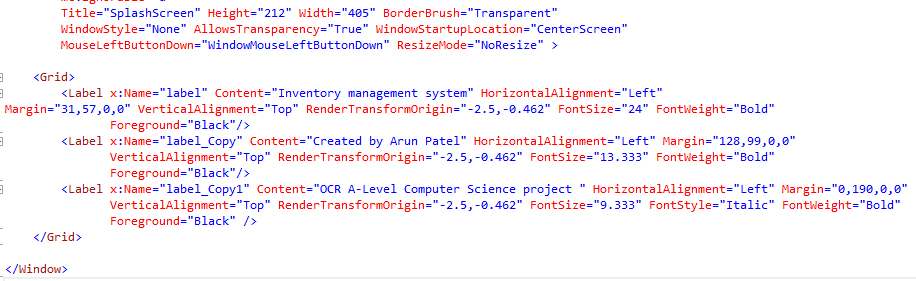
## 1 Interface

I will be replicating my interface designs using the C# programming language and the Windows Presentation Foundation, which makes use of XAML (Extensible Application Markup Language) to design interfaces.

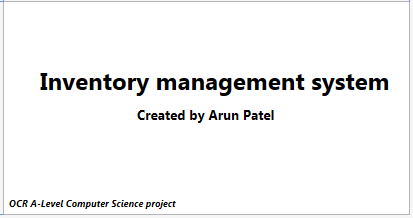
As described in the design section, the interface will feature three main windows, the splash window, login window and the main window. To start with I began designing the splash screen.

The XAML for the initial splash screen design was the following:

### 1.1 Splash Screen



This produced the following,



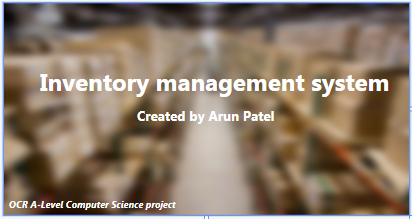
I then added a background to the splash screen and by using the background property on the grid.



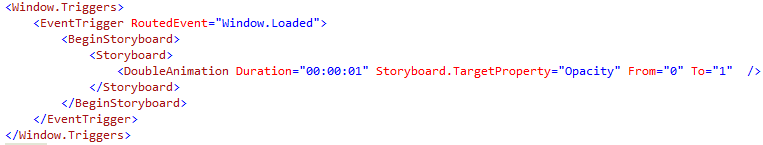
The image, inventoryBLUR, is an image I editing in Photoshop of a warehouse with a Gaussian blur added. I also changed the colour of the text on the splash screen to white by making use of this property on each label. I could have made use of styling but chose not to since there are only 3 labels.



These changes produced the following result.



The following XAML made the splash screen fade into the screen rather than just appear.



The following XAML is the finalised splash screen window.

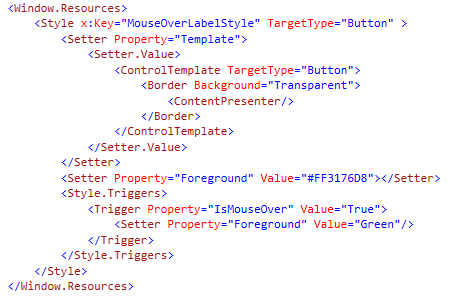


### 1.2 Login window

Now that I have designed my splash screen I will now design my login screen. The following is the XAML code used to create the window.

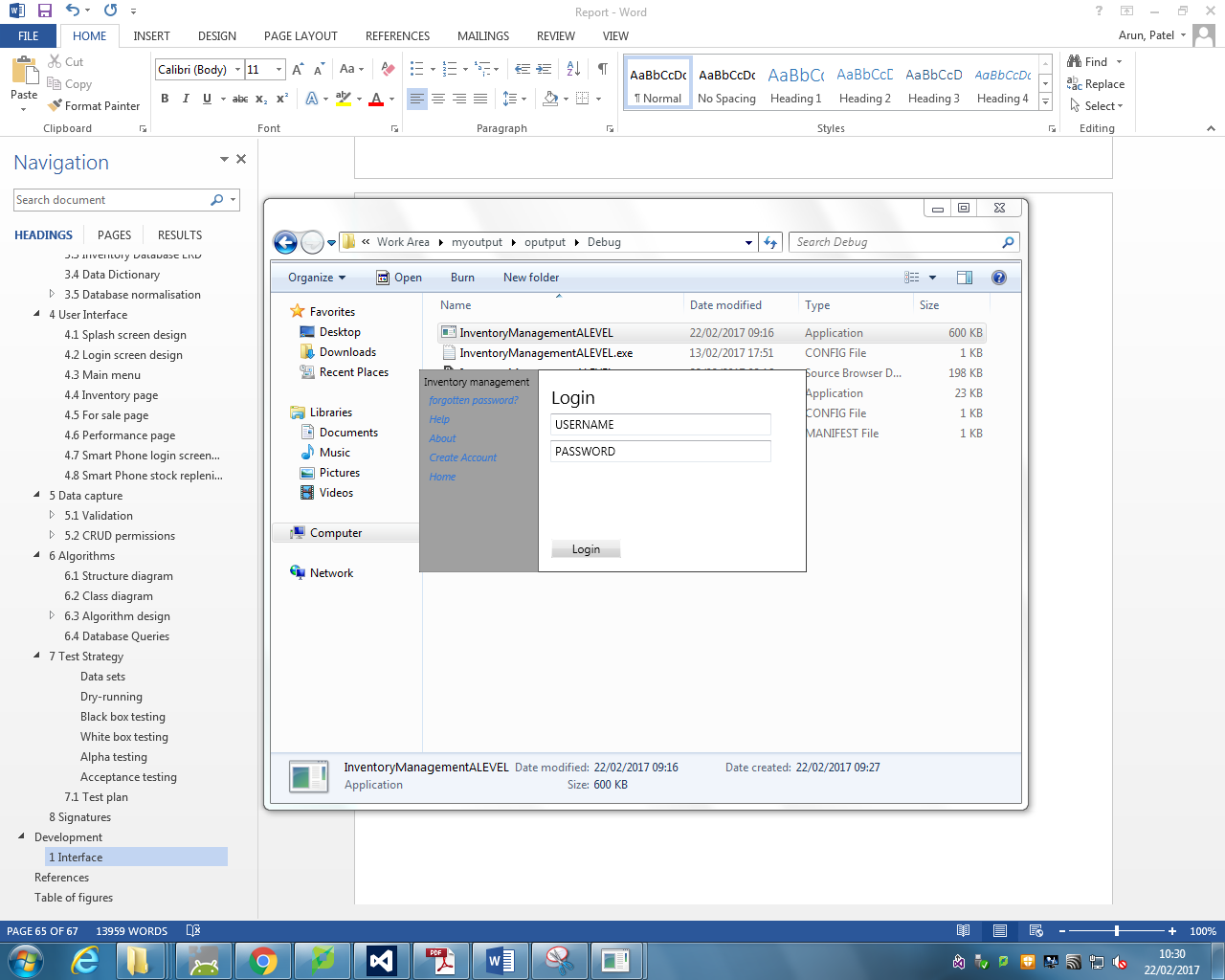


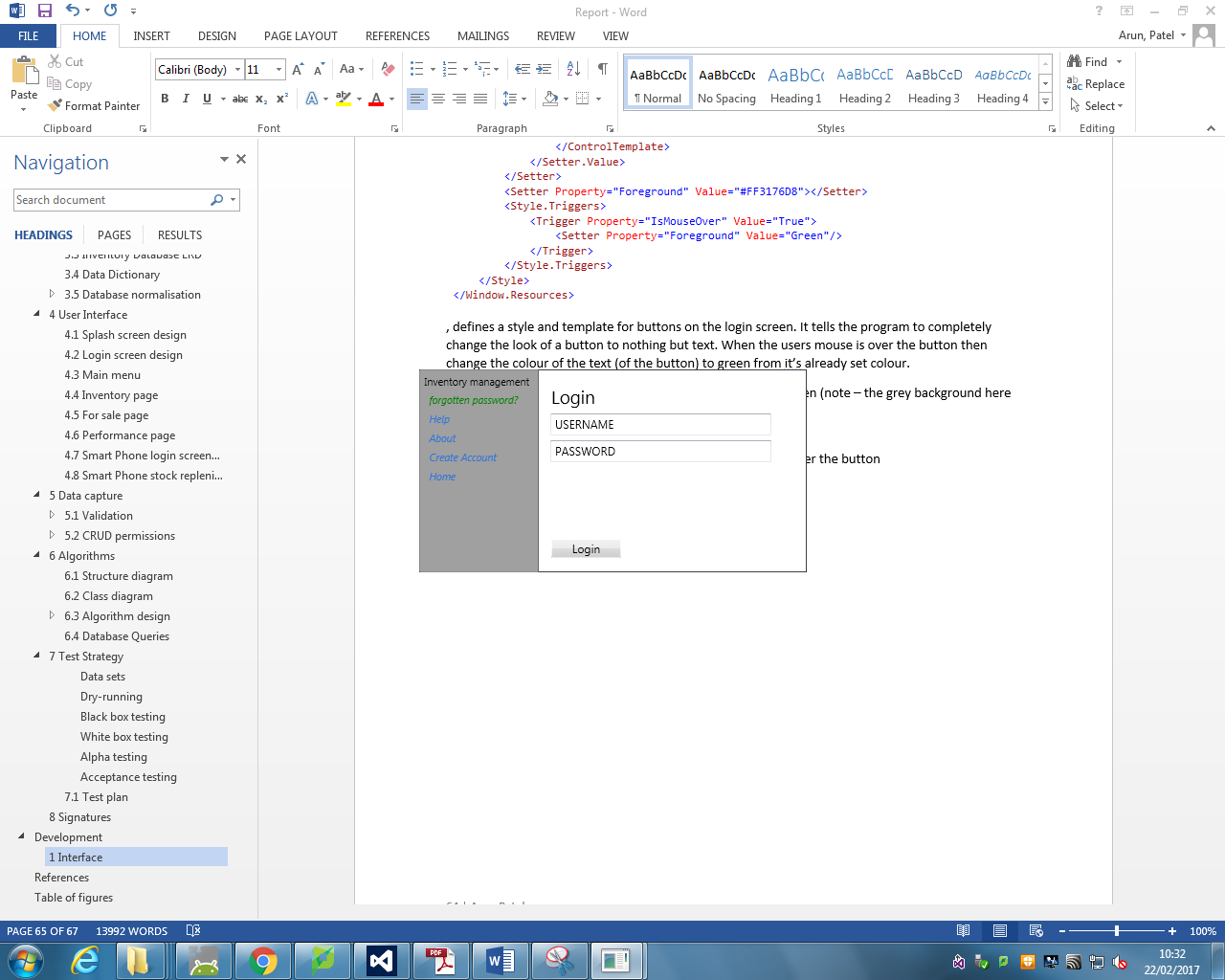
This section of the XAML code



, defines a style and template for buttons on the login screen. It tells the program to completely change the look of a button to nothing but text. When the users mouse is over the button then change the colour of the text (of the button) to green from its set colour.

This produces the following result on a button in the login screen (note – the grey background here is independent of the forgotten password button):

 When the user’s mouse is not over the button

 When the users mouse if over the button.

The following XAML sets up the forgotten password button and assigns the created style to it.

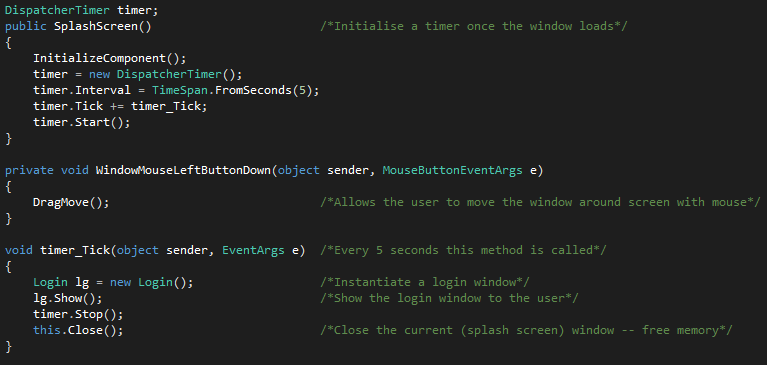


The XAML above creates a black border around the edges of the login window (since I defined the window to have no Window Style). The buttons on the login window are placed in a stack panel, so they are appear sequentially from top to bottom.

The finalised logins screen is the following:



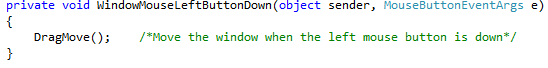
In order to make the login screen appear once the splash screen has closed and logic for moving the splash screen, I used the following code in the c# (.cs) file associated with the splash screen XAML file.



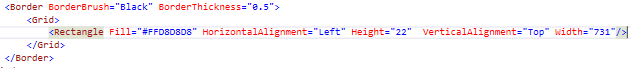
### 1.3 Main window

The main window has the following XAML.

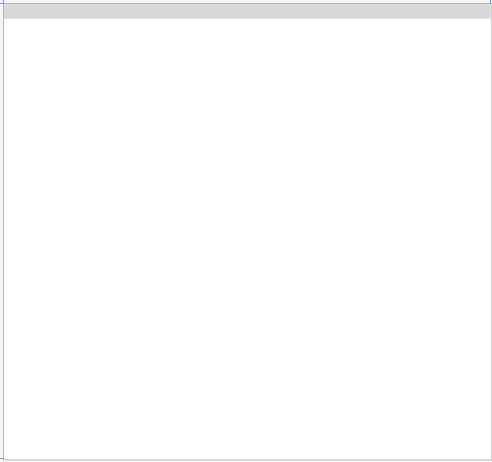
This line of XAML, creates a borderless plain window that starts in the centre of the screen. It has a mouse left button down event, which provides functionality for moving the window around the screen. The c# code for handling this event is shown below.



My main windows grid contains the following. It creates a border around the window and grey toolbar at the top of the screen.



Here is the result that this produces:



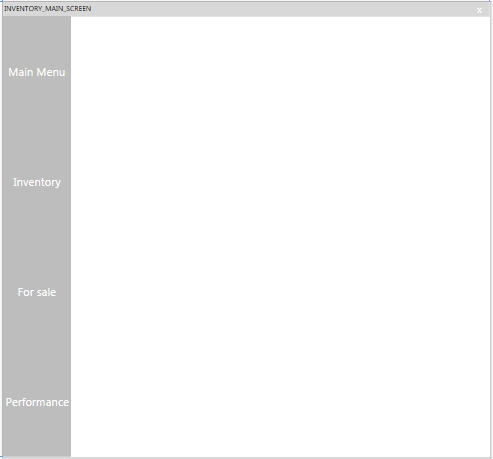
I have also created a template and style for two types of buttons the main window.



This creates buttons that are illuminated blue when hovered over and made even darker when clicked.



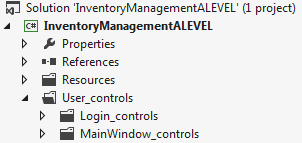
This creates the following window.



### 1.4 Login window user controls

In order to dynamically change the content of an interface such as the login window, I will be hiding and showing user controls onto the window itself. The benefit of this is that user will not be exposed to numerous amounts of popup windows but will easily navigate through a single window with information being rendered onto it dynamically.

In order to keep my work organised, I have made use of folders within my visual studio solution.



User controls for each window are stored in their corresponding folders.

Here resources such as images and music files are stored

#### 1.4.1 Main login user control

In order for the user control to fit into the required space on the login screen, I have used the following dimensions, shown below in XAML code.



This means that all the content in the user control I created will only be shown in the white space of the login screen (the right hand side of the image shown below).

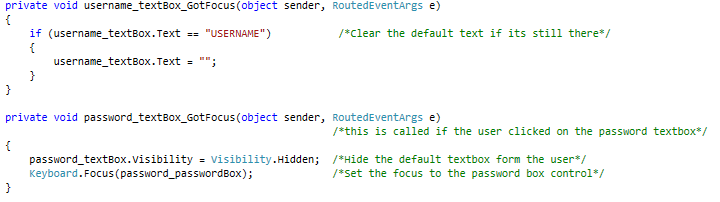


The content of the user control can be easily be created with the following XAML.

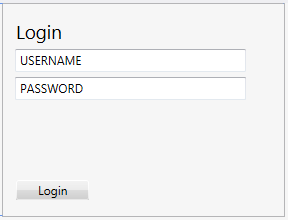


The user control contains a textbox control, called “password\_textbox” on top of a password box control called “password\_passwordbox”. This is due to the fact that the password box control, by default, doesn’t allow for the password box to display a default value. To get around this I used a text box control to display a default message, “password”. Once the user has clicked on the password text box I have set it clear its value, disable itself and set the users focus on to the password box control. The error label is used to display an error message when the user enters an incorrect login and password combination.

The c# code for these events are shown below.



The following image is of the finalised user control.



In order to draw this user control onto the main login window I have included the following line of XAML inside the main login window grid.



This allows me to dynamically draw user controls onto login windows main screen by changing the value of the content control.

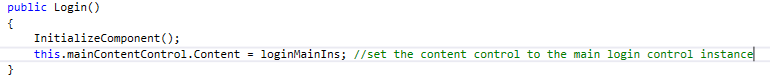
In the login screens c# code, I created a new global instance of the user control I have just created.



The benefit of making this a global variable, instead of instantiating the user control each time its needed , is that it is far more efficient , less resources are needed to create the instance as it is only instantiated once at the runtime.

Furthermore, the data entered in the user control will remain, for example if the user enters half of their username and then clicks on the about button and then returns to the main page again, their username will remain. However, if I chose to instantiate the user control each time it’s needed, the user’s half entered username would not remain, due the user viewing a different instance of the user control.

I have chosen for the user control I have just created to be shown when the login screen first loads and when the user clicks on the home button on the login main window. Below is the c# code to allow this to happen.

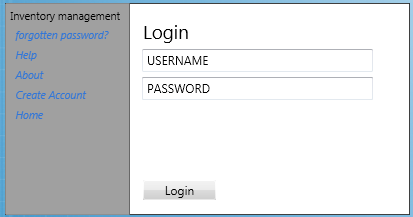


This is placed in the constructor of the login windows class.



This section of code does the same as the above but is called when the user clicks on the home button.

When running, this is the result produced when the user control is rendered onto the login screen.



#### 1.4.2 Create account user control

Following on from the previous user control I created, this user control will have the same dimensions as shown below.

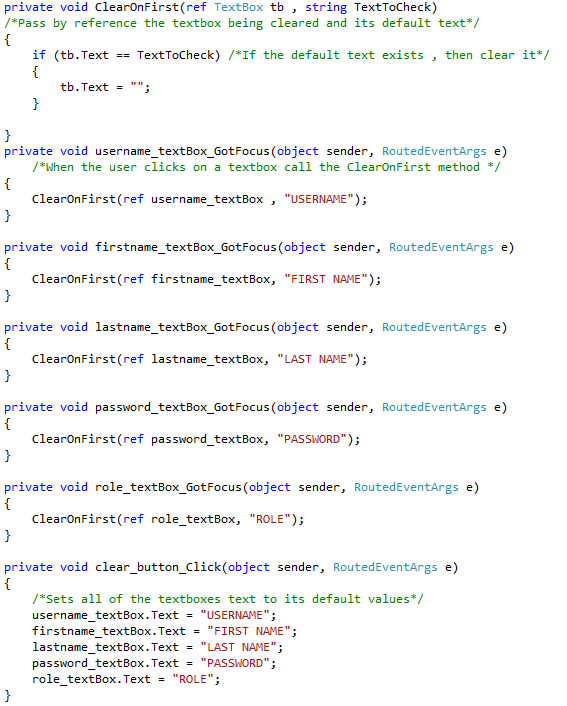


The content of the user control is shown below in XAML,



Each textbox has its own got focus event, used to detect when the user has clicked on the textbox, so that I can clear its contents if the textbox contains default text (as previously done with the previous user control). The clear button resets each field to their default text.

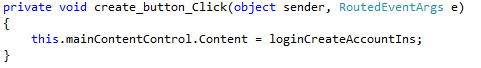
The c# code associated with this XAML is shown below:



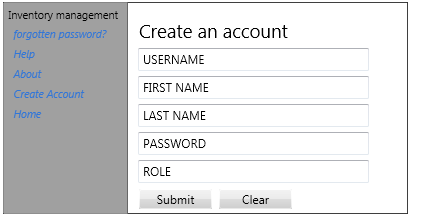
In order to have this user control render itself onto the login window, I created a new global variable instance of the user control in the main login windows C# file, as shown below:



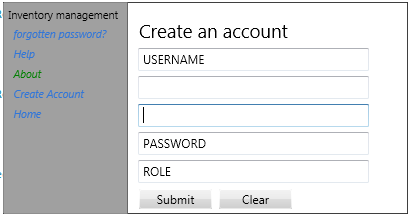
When the user clicks on the create account button, the program executes the following code, causing the user control to show:



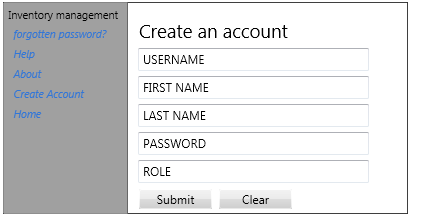
The image below shows the login screen displaying the user control I have just created.



When the user clicks on the textbox, the text is cleared, as shown below.



Clicking the clear button, causes the textboxes text to reset, as shown below.

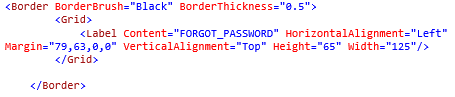


#### 1.4.3 Forgotten password user control

As with the other user controls, this one will have the following dimensions:



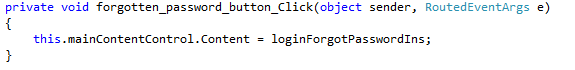
The XAML for this user control is the following:



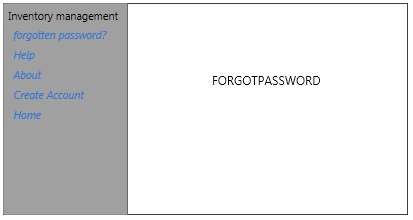
A global variable instance was created in the login windows main C# file:



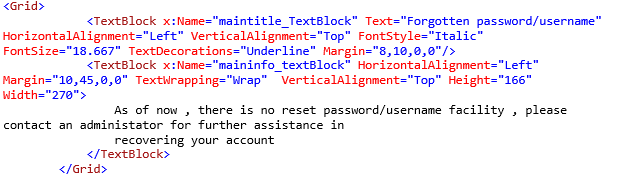
The following code, shows the user control when the user clicks on the forgotten password button:



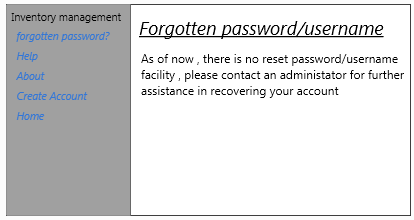
The image below is of the program running and displaying the forgotten password user control.



I then updated the content of grid with the following:



This produces the following:



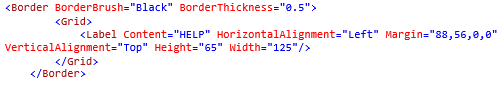
#### 1.4.4 Login help user control

As with the forgotten password user control, I will be creating a blank user control to start with

Its dimensions are declared as the following:



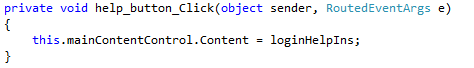
Its contents are as follows:



The global variable instance of this user control in the login screens main C# file is shown below



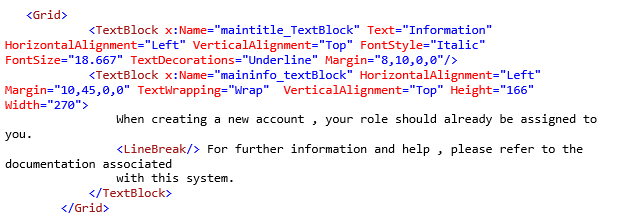
This user control is shown when the user clicks on the help button



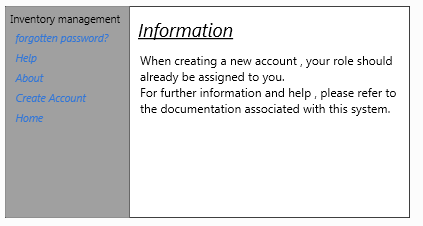
The image below shows the user control being displayed when the program is running:



I then updated the content of grid with the following:



This produced the following result:

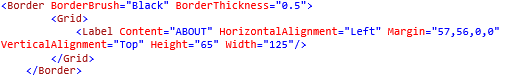


#### 1.4.5 Login about user control

The dimensions of this user control are declared as the following:



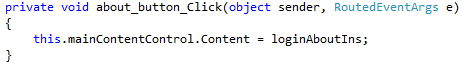
The contents are as follows:



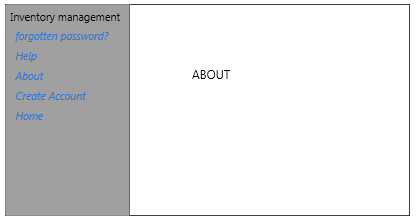
The global variable instance of this user control in the login screens main C# file is shown below



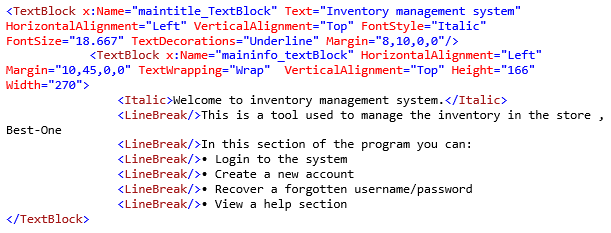
This user control is shown when the user clicks on the about button



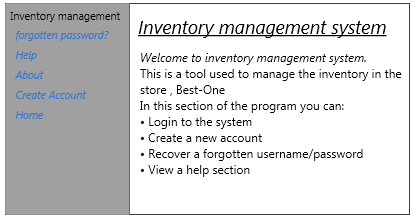
The image below shows the user control being displayed when the program is running:



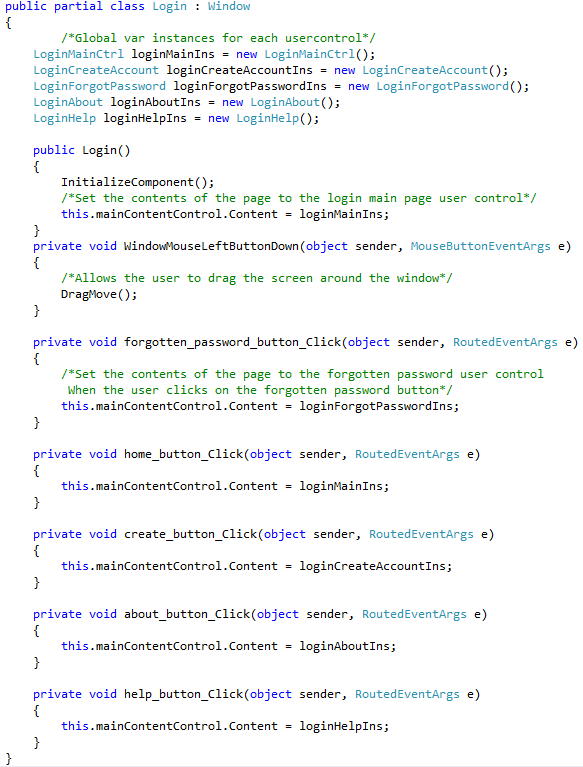
I then updated the content of grid with the following:



This produced the following result:



The main login screen C# file now looks like the following:



In order to access the user controls folder within code, I used the following import:



### 1.5 Main window user controls

Similarly to the login windows user controls, I will store the main windows user controls in the main window controls, which is stored in the user controls folder.

#### 1.5.1 Main page user control

This user control is displayed when the main window firsts loads.

Its dimensions are as follows:

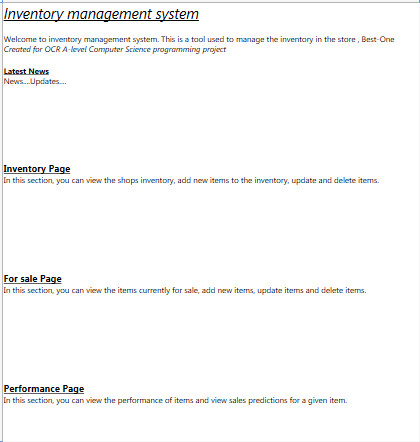


Its contents are as follows:



As shown above, the main page user control consists of a series of text blocks used to display information. As of now, the main page will not contain anything else.

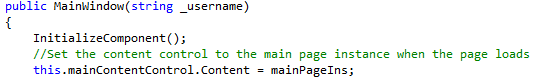
The following is shown as a result of the XAML just created.



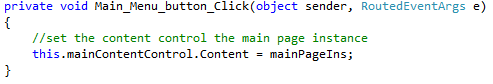
In order to have this user control be displayed on the main window, in the main window C# file I created a global variable of an instance of this user control:



To have this user control show when the window loads I have added the following code to the main windows constructor:



When the user clicks on the main menu button on the main window, the user control should be shown, therefor I have added the following code to the button clicked event method for the main menu button:



The following is an image of the program running showing the main page:

{

}

#### 1.5.2 Main Inventory user control

This user control is the page which will display the main inventory table to the screen and allow the user to manipulate this table.

### 1.6 Prototype

# References

Anonymous, 2016. *Acctivate Inventory Management Software.* [Online]   
Available at: http://www.softwareadvice.com/uk/inventory-management/acctivate-profile/  
[Accessed 21 September 2016].

Anonymous, 2016. *inFlow Inventory Software.* [Online]   
Available at: http://www.softwareadvice.com/uk/inventory-management/inflow-inventory-profile/  
[Accessed 21 September 2016].

Azure, M., 2016. *SQL Database.* [Online]   
Available at: https://azure.microsoft.com/en-gb/services/sql-database/  
[Accessed 4 January 2017].

ERS, 2016. *ERS.* [Online]   
Available at: https://www.ers-online.co.uk/o5977/ers-stockcount-kit1-ers-stockcount-barcode-stock-taking-solution  
[Accessed 27 November 2016].

Fletcher, M., 2016. *Introduction to Network Programming.* [Online]   
Available at: http://www.webbasedprogramming.com/Java-Unleashed-Second-Edition/f23-1.gif  
[Accessed 21 September 2016].

Microsoft, 2015. *Windows 10.* [Online]   
Available at: https://www.microsoft.com/en-gb/windows/windows-10-specifications  
[Accessed 2 November 2016].

Microsoft, 2016. *Supported Query Types (Visual Database Tools).* [Online]   
Available at: https://technet.microsoft.com/en-us/library/ms188316(v=sql.110).aspx  
[Accessed 12 December 2016].

Microsoft, n.d. *Office Word,* s.l.: s.n.

Templates, A., 2017. *Access Inventory Management Control Templates,* s.l.: s.n.

# Table of figures

[Figure 1 8](#_Toc469316654)

[Figure 2 9](#_Toc469316655)

[Figure 3 26](file:///E:\OCR%20A%20Level%20Computer%20Science%20project\Report.docx#_Toc469316656)

[Figure 4 26](file:///E:\OCR%20A%20Level%20Computer%20Science%20project\Report.docx#_Toc469316657)

[Figure 5 27](file:///E:\OCR%20A%20Level%20Computer%20Science%20project\Report.docx#_Toc469316658)

[Figure 6 28](file:///E:\OCR%20A%20Level%20Computer%20Science%20project\Report.docx#_Toc469316659)

[Figure 7 28](file:///E:\OCR%20A%20Level%20Computer%20Science%20project\Report.docx#_Toc469316660)

[Figure 8 29](file:///E:\OCR%20A%20Level%20Computer%20Science%20project\Report.docx#_Toc469316661)

[Figure 9 30](file:///E:\OCR%20A%20Level%20Computer%20Science%20project\Report.docx#_Toc469316662)

[Figure 10 31](file:///E:\OCR%20A%20Level%20Computer%20Science%20project\Report.docx#_Toc469316663)

[Figure 11 32](file:///E:\OCR%20A%20Level%20Computer%20Science%20project\Report.docx#_Toc469316664)

[Figure 12 33](file:///E:\OCR%20A%20Level%20Computer%20Science%20project\Report.docx#_Toc469316665)

[Figure 13 34](file:///E:\OCR%20A%20Level%20Computer%20Science%20project\Report.docx#_Toc469316666)

[Figure 14 35](file:///E:\OCR%20A%20Level%20Computer%20Science%20project\Report.docx#_Toc469316667)