Av: Magnus Andersson
Version: 2

1. Antaganden

När alla varor skulle ha ett unikt ID, så antog jag att man skulle generera det själv och lägga in en id på varje vara som fanns sedan tidigare i CSV filen. De varor som sedan läggs till som nya produkter måste sedan alltid ha en unik id annars kommer man inte kunna lägga till produkten. Antog också att ID alltid skulle vara ett heltal.

Då alla varor också skulle kunna finnas på lager, så antog jag att man behövde en extra data i CSV filen som jag kallade antal. Sedan gav jag de varor som redan fanns något slumpmässigt antal och om man skapar en ny produkt, men inte specificerar antalet så kommer antalet automatiskt bli 0.

Namn och pris skulle vara obligatoriska, så då antog jag att man inte låter någon skapa en produkt där dessa fält är tomma. Som nämnt antog jag att id som lagts till också behövde vara obligatoriskt. Så om man inte skriver något i dessa tre fält får man inte skapa en ny produkt.

I version 2 av programmet så skulle en sökfunktion finnas med. Denna antog jag bara skulle finnas i butikvyn även om den tekniskt sätt hade kunnat finnas på båda. Men då specen inte specificerade vilken vy den skulle finnas på valde jag att lägga den på enbart butiksidan. Antog också att följande attribut var intressanta att kunna söka efter: Namn, Id och Författare.

Version 2 skulle också ha någon typ av återköpfunktion. Då det inte specificeras hur detta skulle utföras, så antog jag att om man klickar på en produkt i butiken så får man alternativet att klicka på en knapp som heter retur som returnerar tillbaka en styck av vald produkt. Förutsättningen är att varan finns kvar i lagret, då den kan ha tagits bort.

2. Översikt

Programmet består av ett fönster som huvudsakligen har två vyer. En vy för en butiksida och en annan för en lagersida. Man startar alltid på vyn från butiksidan och den visar upp en lista med produkter som finns i butiken på vänster sida av fönstret.

Ovanför listan finns en sökbar med en sökknapp bredvid den. Med denna kan man söka efter en specifik produkt genom att skriva dess namn, id och författare och listan under blir uppdaterad för att visa det man sökte efter.

Till höger om butiks vyn finns också en annan lista, men denna är tom till att börja med tills man klickar på en produkt i den vänstra listan och väljer den knapp som finns mellan listorna. Denna knapp har två pilar till höger på sig och tar den valde produkten och lägga in den i listan till höger. Detta kan göras med så många produkter man vill, så länge de finns på lager.

Om det är så att man vill lägga tillbaka en vald produkt kan man använda knappen som är mellan listorna som har två pilar till vänster. Denna knapp tar tillbaka den produkt som du inte längre vill ha och lägger tillbaka den i lagret.

Om man känner sig färdig och har valt de produkter man vill ha kan man klicka på en knapp som heter köp produkter som kommer lagra de produkter som valts, så att man kan skriva ut

det till ett kvitto. Antalet produkter som köpts kommer uppdateras i lagret så fort man köpt produkterna.
Sist på butiksidan finns det en retur knapp. Denna knapp returnerar en viss vara om man vill det. För att använda den måste man först klicka på den produkt som önskas returneras och sedan klicka på retur knappen, så ökar antalet för den produkten med ett.

Man kan högts upp till vänster byta vy på programmet till en lagersida. Denna sida har en lista på vänster sida precis som butiksidan som visar alla produkter som finns i lager.

Klickar man på en produkt i listan får man information om denna produkt i textboxarna till höger om listan. Om det är så att man önskar uppdatera en produkt så kan man ändra informationen i textboxarna för den produkt man valt och sedan klicka på lagra knappen under textboxarna.

Om man önskar lägga till en ny produkt, så kan man skriva in informationen som man vill att den ska ha och klicka på lagra om ingen produkt är vald sedan innan. Om man valt en produkt och önskar lägga till en ny produkt kan man klicka på knappen ny produkt för att rensa textboxarna från information och så kan man skriva in sitt eget för den nya produkten.

Om man önskar ta bort en produkt, så klickar man först på den produkt man vill ta bort och klickar sedan på knappen ta bort. Denna tar då bort produkten, men om det är så att produkt antalet inte är noll för den valda produkten kommer en varning komma upp som frågar om man verkligen vill ta bort produkten eller inte.

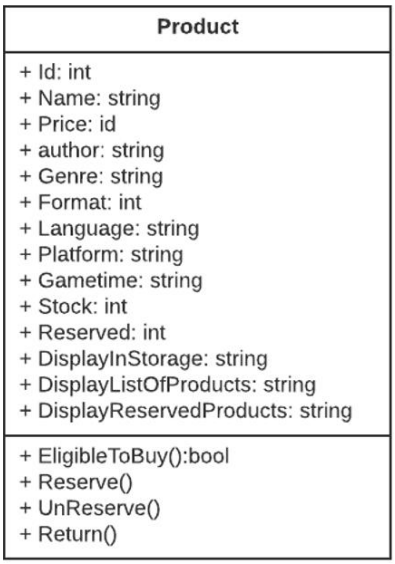
Både listan i butiksidan och den i lagersidan uppdateras samtidigt. Så om något ändras på butiksidan kommer lagersidan uppdateras direkt och vise versa.

3. Detaljerad beskrivning

Programmet är uppbyggt med en simpel backend-frontend design. Där hjälpfunktioner för backend finns med i klasserna Product och Storage och allt som har med frontend att göra ligger i Form1 klassen.

Klass Product:

En klass för hantering av alla egenskaper som en produkt ska ha. Klassen hjälper till att lagra och ta hand om produkternas information. Den har också några metoder som hjälper till för att få ut rätt antal produkter i alla listor.



Properties:

- public Id: int – Ett heltal som representerar id för produkten
- public Name: string – En textsträng som representerar namnet för produkten
- public Price: id – Ett heltal som representerar priset för produkten
- public Author: string – En textsträng som representerar författaren för produkten
- public Genre: string – En textsträng som representerar genren för produkten
- public Format: int – Ett heltal som representerar formatet för produkten
- public Language: string – En textsträng som representerar språket för produkten
- public Platform: string – En textsträng som representerar plattformen för produkten
- public Gametime: string – En textsträng som representerar speltiden för produkten
- public Stock: int – Ett heltal som representerar antalet för produkten
- public Reserved: int – Ett heltal som representerar reservation för produkten
- public DisplayInStorage: string – En textsträng som representerar visningen av produkter i storage
- public DisplayListOfProducts: string – En textsträng som representerar visningen av produkter i butikslistan
- public DisplayReservedProducts: string – En textsträng som representerar visningen av produkter i listan av valda produkter

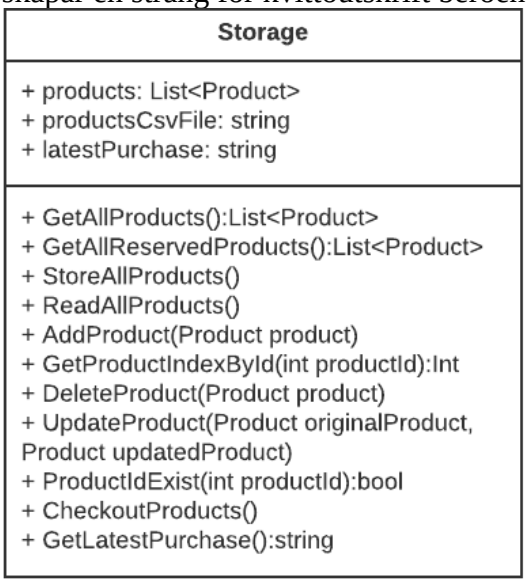
Metoder:

- public EligibleToBuy():bool – Kollar om man kan köpa eller inte beroende på hur många som finns och hur många som reserverats
- public Reserve() – Ökar antalet på variablen reserved med ett
- public UnReserve() – Minskar antalet på variablen reserved med ett
- public Return() – Ökar antalet på variabeln stock med ett

Klass Storage:

En klass för hantering av allt som ska hända i lagret. Läser in och skriver in till en CSV fil med produkter. Skapar upp en lista med alla produkter som finns och den uppdateras/ändras med hjälp av olika metoder som finns inne i klassen. Inne i denna klass finns det metoder för att hämta, lagra, läsa, lägga till, ta bort, uppdatera, reservera produkter och lagra/hämta kvittoinformation. Finns även metoder att få en index beroende på id, en check på om id finns

sedan tidigare, en checkout som anropas när man köper produkterna och en som tar emot och skapar en sträng för kvittoutskrift beroende på det senaste som köpts.



Variabler:

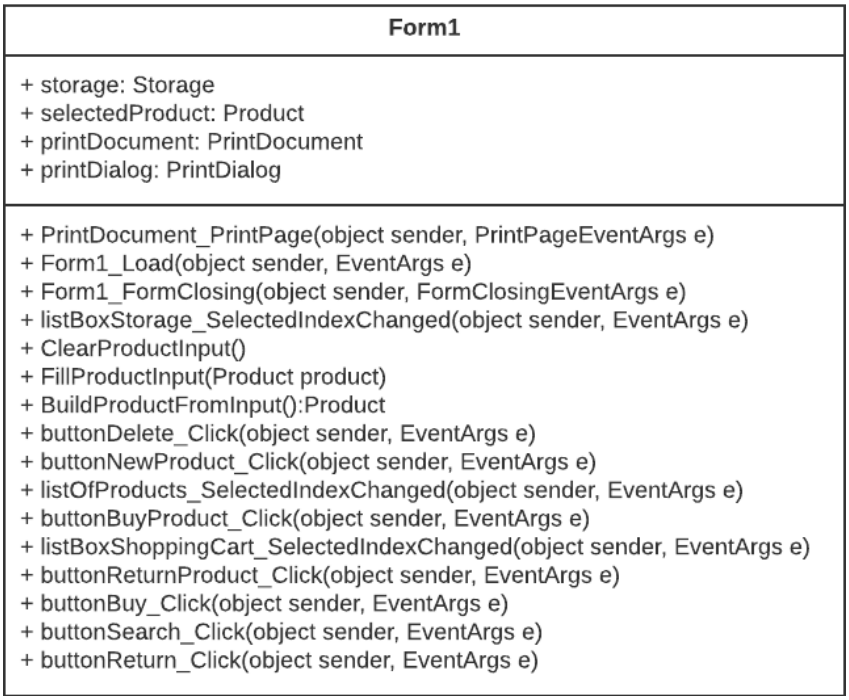
- public products: List<Product> – En tom lista som ska innehålla alla produkter
- public productsCsvFile: string – En sträng som länkar till en CSV fil med produkter
- public latestPurchase: string – En tom sträng som ska innehålla det senaste köpet av produkter för att användas till kvittoutskrift

Metoder:

- public GetAllProducts(string filter):List<Product> – Returnerar en lista av alla produkter. Om man skriver in en textsträng för en produkt kommer den filtrera mot namn, id eller författare. Filter används för sökning.
- public GetAllReservedProducts():List<Product> – Får alla reserverade produkter och returnerar dem. Används för visning i listan av valda produkter.
- public StoreAllProducts() – Lagrar alla produkter som finns i listan products till CSV filen
- public ReadAllProducts() – Läser in alla produkter från CSV in till listan products
- public AddProduct(Product product) – Lägger till en produkt till listan products
- public GetProductIndexById(int productId):Int – Får produktens index(posititon) i listan products med hjälp av id på produkten
- public DeleteProduct(Product product) – Tar bort en produkt från listan products
- public UpdateProduct(Product originalProduct, Product updatedProduct) – Uppdaterar produktinformation för en produkt
- public ProductIdExist(int productId):bool – Kollar om produkt ID finns sedan innan
- public CheckoutProducts() – Checkar ut och skriver upp en sträng med det som har köpts
- public GetLatestPurchase():string – Returnerar text översikt på det senaste köpet som används för utskrift

Klass Form1:

En klass för uppbyggnad av gränssnittet. Allt som har med frontend att göra händer inne i denna klass och den får information genom att den använder sig av klasserna Storage och Product.



Variabler:

- storage: Storage – Variabel som gör att Form1 klassen kan använda metoder och properties inne i Storage klassen
- selectedProduct: Product – Variabel som gör att Form1 klassen kan hålla reda på vilken produkt som är vald och då kan man använda metoder och variabler inne i Product klassen
- printDocument: PrintDocument – Variabel för att innehåller en referens till ett PrintDocument objekt för att göra det möjligt att skriva ut
- printDialog: PrintDialog – Variabel för en dialogbox för att printa

Metoder:

- private PrintDocument_PrintPage(object sender, PrintPageEventArgs e) – Skapar en text som har med det du senast köpt. Används för utskrift av kvitto
- private Form1_Load(object sender, EventArgs e) – Anropar funktionen som visar upp produkterna som är lagrade i lagret när gränssnittet laddar in
- private DisplayProductsFromStorage() – Funktion som uppdaterar visning av produkter i lager, butik och valda produkter. Denna funktionen anropas från flera olika platser i programmet när man har behov för att uppdatera visning av produkter
- private Form1_FormClosing(object sender, FormClosingEventArgs e) – Lagrar produkter till lagret om man stänger ner programmet. Använder hjälp metod i storage för detta
- private listBoxStorage_SelectedIndexChanged(object sender, EventArgs e) – Händer när man klickar på en produkt i listBoxStorage. Sätter selectedProduct till den produkt man valt.
- private ClearProductInput() – Rensar alla produkter i input boxarna
- private FillProductInput(Product product) – Fyller i alla produkter i input boxarna för att kunna editera en produkt

- private BuildProductFromInput():Product – Kollar vad som ligger i input boxarna och bygger upp en ny produkt
- private buttonDelete_Click(object sender, EventArgs e) – Knapp för att ta bort en produkt
- private buttonUpdateProduct_Click(object sender, EventArgs e) – Knapp för att uppdatera och lagra en ny produkt
- private buttonNewProduct_Click(object sender, EventArgs e) – Knapp som rensar input boxar så att man kan skriva in saker i input boxarna för en ny produkt
- private listOfProducts_SelectedIndexChanged(object sender, EventArgs e) – Händer när man klickar på en produkt i listOfProducts. Sätter selectedProduct till den produkt man valt.
- private buttonBuyProduct_Click(object sender, EventArgs e) – Knapp som ser om selectedProduct är vald och utför en reservation på produkten
- private listBoxShoppingCart_SelectedIndexChanged(object sender, EventArgs e) – Händer när man klickar på en produkt i listBoxShoppingCart. Sätter selectedProduct till den produkt man valt.
- private buttonReturnProduct_Click(object sender, EventArgs e) – Knapp för att ta tillbaka en produkt som man köpt
- private buttonBuy_Click(object sender, EventArgs e) – Knapp för att köpa produkterna som har reserverats.
- private buttonSearch_Click(object sender, EventArgs e) – Knapp som söker efter det som är inskrivet i sökfältet
- private buttonReturn_Click(object sender, EventArgs e) – Knapp som ser om selectedProduct är vald och tar bort en reservation på produkten

4. Problem

Första versionen med visning av produkter i en listbox gjordes genom att bygga upp textsträngar. Det gjorde det svårt att hitta vilken produkt man hade valt när man valde i listan. Provade att jobba med både vald position och att parsa innehåll i text för att ta reda på vilken produkt man jobbade med. I vald lösning läggar man in produkterna direkt i listboxen. Det som var lite lurigt var att få till visning av produkten i listan. Detta löstes med 3 olika egenskaper som kunde visa olika texter i de 3 olika listboxarna lager, butik och korg.

Ett problem var att använda produkter med långa titlar/namn och få de att få plats. Detta löste jag genom att använda en metod som heter Truncate som jag hittade på stackoverflow. Detta är en extension metod som kan användas på strängar för att bestämma hur långa de får lov att vara innan det skriv ut ’…’ bakom dom, så att de inte blir för långa.

Utskrift var ett stort problem på hur man skulle få det till. Hittade en bloggpost som jag försökte använda och fick slutligen till att skriva ut en textsträng.

Det var också klurig att förstå konceptet med inläsning av CSV filer och att jobba med den.

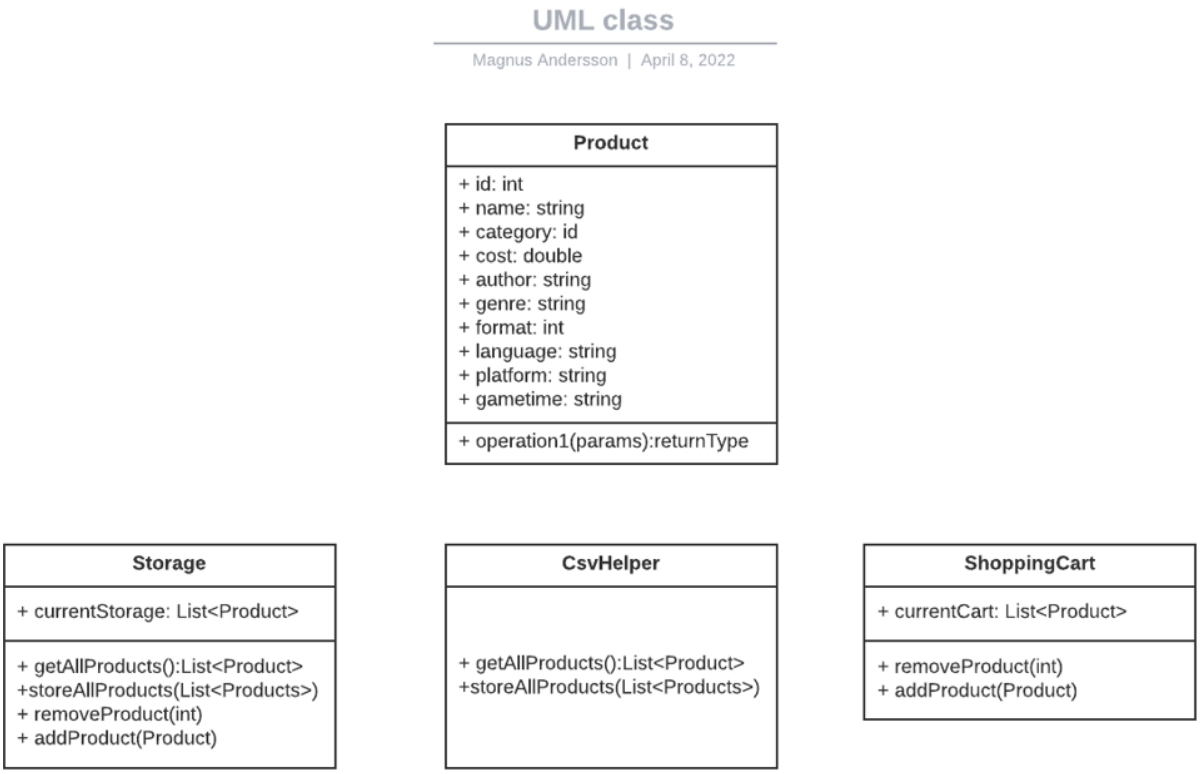
5. Sammanfattning

Sammanfattningsvis så har projektet gått ganska bra, men med några hinder på vägen. Insåg fort att jag skulle behöva mycket mer variabler och metoder än vad som tänktes från första början. Blev en del omskrivning av kod på hur jag skulle bäst och enklast kunna få ut vilket index man var på, så att jag kunde använda den. Blev till slut en funktion som fick ut vilket index man är på med hjälp av produktens id. Det var en del saker som inte fanns i specen som

man var tvungen att fixa och tänka till själv för att lösa, vilket var svårt då man alltid har i baktanke att man inte gör på bästa sätt. Har kanske lagt lite mycket funktionalitet i storage klassen, då den hanterar både butik och lager. Den hade kanske kunnat skiljas på, men eftersom båda butik och lager jobbar mot samma produktlista blev det enklast. Vet inte direkt hur lång tid projektet har tagit per moment, men skulle gissa på som helhet att det tagit mig runt 50-70 timmar att få allt att fungera som jag ville.

Bilagor

Start Diagram:



Avslutande Diagram: