# Projektarbete – WebForms/SQL

Ni har fått i uppdrag av en kund att skapa en webbshop där kunden kan sälja sina produkter. I dagsläget säljer kunden enbart produkterna i sin fysiska butik men känner att det är dags att börja använda sig av webben för att öka sin försäljning.

Uppdraget har två delar – delvis att skapa en databas för kundens produkter samt att skapa en webbshop där produkterna kan köpas.

Kunden har ingen som helst koll på design så ni har fria händer att designa webbshoppen som ni vill med utgångspunkt från de krav som ställs. Kunden säljer precis allting i sin affär så ni får själva lägga in några exempelprodukter som ska vara på plats från början. Var kreativa.

## Allmänna krav

Webbshoppen skall skapas med *ASP.NET WebForms, HTML, CSS* och *JavaScript/jQuery*. Senaste versionen av *Chrome* kommer användas för att testa funktionaliteten.

Gällande *JavaScript/jQuery* – känn er inte tvingade att lägga in några script bara ”för att”. Behövs det inga script för att få er sida att fungera enligt era önskemål så är det okej.

*Bootstrap* får användas om så önskas.

På databassidan ska *SQL Server* användas i kombination med *Entity Framework*. *Database First* ska användas, det vill säga ni måste skapa databasen själva på det vis vi gjort tidigare i kursen.

Utseendet på sidan är egentligen det minst viktiga. Förväxla dock inte det med att sidan ska ha en vettig struktur. En användare av sidan borde intuitivt förstå och smidigt kunna nå webbshoppens olika delar. Det allra viktigaste är att det tekniska fungerar och att inga oväntade fel uppstår.

Ni får själva till stor del ta beslut om namn och datatyper på era tabeller, kolumner, klasser etc. Tänk igenom era beslut noga!

Nedan följer de specifika krav som ställs på webb- samt databas-delen.

# WebForms

## Krav för G – godkänt

Följande funktionalitet/innehåll skall finnas på webbshopen:

* Inloggning
* Registrera användare
* Lista produkter
* Visa detaljer om en produkt – med möjlighet att lägga produkten i en kundvagn
* Visa kundvagn (samt genomföra köp)
* Information/kontakt-sida
* Mitt konto-sida
* Utloggning

### Inloggning

* **E-postadress** och **lösenord** ska anges för att logga in.
* Användaren ska kunna navigera till en sida för att kunna registrera en ny användare.
* Vid lyckad inloggning ska användaren hamna på *Lista produkter*-sidan.
* **OBS**! Användarna MÅSTE logga in för att komma åt webbshoppen överhuvudtaget. Er uppdragsgivare vill ha full kontroll över vilka som använder webbshoppen i början – så **inloggning** **ska krävas.**

### Registrera användare

När man registrerar användare ska minst följande anges:

* **Förnamn**
* **Efternamn**
* **Ålder**
  + Endast heltal, påtvinga också vettiga värden.
* **Email**
  + Måste vara en korrekt formad e-postadress, t.ex. *example@domain.com.*
  + Måste vara unik, två användare får inte ha samma e-postadress – ge användaren ett meddelande om att e-postadressen redan finns i systemet om en existerande anges.
* **Lösenord**
  + Lösenordet får inte synas i textboxen när man skriver, inmatning ska döljas.
  + På grund av ovanstående måste användaren **upprepa lösenordet**i en annan textbox*.*
* **Skicka erbjudanden** (*checkbox*, ska indikera om användaren accepterar e-postutskick).

Alla dessa egenskaper ska kunna anges på lämpligt sätt när man registrerar en ny användare. Uppgifterna ska sparas i databasen. Alla egenskaper måste anges, inga fält får lämnas tomma. **Validera** all inmatning.

*Avgränsningar*:

Det går bra att lagra användarens lösenord i klartext i databasen. Ingen hashning/kryptering av lösenord behöver ske.

## Affärssida

Ni har ganska stor frihet gällande utformningen av webbshop-delen. Snabb repetition av huvuddelarna som bör kunna nås efter lyckad inloggning:

* Lista produkter/startsida
* Produktdetaljer
* Varukorg (samt genomföra köp)
* Information/kontakt-sida
* Mitt konto-sida
* Utloggning

Kunden har hört talats om så kallade ***Master Pages*** och ser helst att sådan används för att ge sidan en grundstruktur.

## Produkter / kategorier

En produkt i systemet ska ha minst följande egenskaper:

* **Namn**
* **Detaljerad beskrivning**
* **Pris**
* **Bild** (sökväg till en bild rekommenderas)
* **Kategori** (för **G** räcker det att en produkt tillhör endast EN kategori, för **VG** ska en produkt kunna ingå i fler kategorier, se detaljer i **VG**-sektionen)

Produkterna i kundens system ska som synes tillhöra en **kategori*.*** I nuläget behöver en kategori endast ha ett **namn**, men eftersom kunden har framtida planer för kategorierna måste en databastabell skapas för kategorierna.

### Lista produkter/startsida

Efter lyckad inloggning ska användaren hamna på den här sidan. Användaren ska här kunna:

* Se en lista på alla kategorier som finns i databasen (en textlänk räcker bra).
* När användaren klickar på en av kategorierna ska en lista visas för användaren som kortfattat presenterar alla produkter i den kategorin. Varje ”listelement” borde åtminstone visa:
  + Produktens namn, pris, samt en liten bild.
* Användaren ska kunna klicka på en produkt för att få fram detaljerad information om just den produkten.

### Produktdetaljer

På sidan som visar produktdetaljer ska all information som finns om produkten presenteras tillsammans med en bild. Här ska även användaren kunna:

* Lägga till produkt i varukorgen. Användaren ska själv få välja ***hur*** ***många*** exemplar som ska läggas till. Specifika krav på antal: minst *1*, max *100*.

### Varukorg

På något sätt borde användaren alltid kunna navigera till sin varukorg. Här ska användaren kunna:

* Se innehållet i varukorgen med **styckpris**, **antal** och **summa** för varje produkt.
  + Man ska inte kunna ha 0 (noll) eller ett negativt antal av en produkt.
* Se totalsumman.
* Ta bort en produkt ur varukorgen.
  + Det behöver inte gå att ändra antal – en borttagningsfunktion räcker.
* Genomföra sitt köp.
  + Här behöver ingenting specifikt göras – det räcker att på något sätt tacka användaren för sin beställning och rensa varukorgen.

*Avgränsningar:*

Varukorgen behöver inte kunna sparas. Om användaren loggar ut eller stänger ner webbläsaren är det okej om varukorgen försvinner.

### Information/kontakt

Den här sidan ska innehålla kort information om företaget samt kontaktadress, e-mail etc.

Inga specifika krav, låt kreativiteten flöda! Hitta på ett företagsnamn och så vidare själva.

### Mitt konto

Presentera här uppgifter om den inloggade användaren. Visa t.ex. fullständigt namn, ålder, e-mail och huruvida man tar emot e-postutskick.

Inga uppgifter behöver kunna förändras. För **VG** ska uppgifterna även kunna uppdateras. Se nedan.

### Utloggning

Användaren ska när som helst kunna logga ut från webbshoppen.

## Krav för VG – väl godkänt

För VG tillkommer några specifika funktioner. I korthet:

* Utökning av kategori-begreppet – precis som förut ska en kategori kunna innehålla flera produkter, men nu borde också en produkt kunna ingå i flera kategorier.
* Uppdatera användaruppgifter
* Orderhistorik

Detaljer följer nedan.

Utöver detta ställs betydligt högre krav på hur koden är strukturerad, namngivning, objektorienteringstänk och så vidare.

### Mer avancerade kategorier

En produkt ska kunna ingå i flera kategorier. Exempelvis borde en Playstation 4 kunna ingå i både en *Skoj-* och en *Elektronik*-kategori. I det här fallet borde alltså en Playstation 4 komma fram oavsett om man väljer att visa *Skoj* eller *Elektronik*-kategorin på webbshoppen.

### Uppdatera användaruppgifter

På *Mitt konto-*sidan ska det gå att uppdatera sina användaruppgifter. Man borde alltså kunna:

* Ändra sitt förnamn, efternamn, ålder, och huruvida man vill få e-postutskick
* Uppdatera sitt lösenord
* Spara ändringar

### Orderhistorik

Någonstans på webbshoppen - presentera orderhistorik för den inloggade användaren. Sortera på datum.

OBS! En order behöver inte innehålla detaljer om vilka produkter som var med. Det räcker med:

* Datum då beställningen gjordes
* Beställningens totalpris

Visa även den **totala summan** för alla beställningar.

Du kommer behöva uppdatera databasen med en tabell som innehåller information om beställningar och spara information när ett köp ”genomförs”.

# Tänk på…

* Försök tänka till hur man som användare vill att en webbshop ska fungera. Tänk på hur andra webbshoppar ser ut och ta inspiration därifrån.
* Få det tekniska att fungera först innan ni eventuellt också gör det snyggt. Fastna inte på designen!
* Försök att göra webbapplikationen stabil – det ska inte krascha för att man råkar mata in exempelvis ”hej” som ålder när man registrerar en användare.

# Databas – SQL

Kunden vill att du skapar ett script för att generera databasen och skapa upp testdata. Minst 5 kategorier och minst 10 produkter ska läggas in från början via scriptet.

Kunden vill även få hjälp att ta fram vissa SQL-frågor för att ta fram viss statistik ur databasen. Detaljer följer nedan.

## Krav för G – godkänt

Tabeller borde finnas för **användare**, **produkt** och **kategori.** En produkt **måste** **tillhöra** **en** **kategori**. Exempel på kategorier kan vara *Skoj* eller *Elektronik*.

### Statistikfrågor

För att kunden ska bli **G**-nöjd behöver kunden få hjälp med att ta fram viss statistik ur databasen. Hjälp kunden genom att ta fram SQL-frågor som tar fram följande:

* Antal produkter databasen innehåller.
* Priset för den billigaste och den dyraste produkten grupperat på kategori. Resultatexempel:

|  |  |  |
| --- | --- | --- |
| Kategorinamn | Billigast | Dyrast |
| Skoj | 100 | 5000 |
| Mat | 10 | 120 |

Lägg dessa SQL-frågor sist i ditt SQL-script som du ska lämna in.

## Krav för VG – väl godkänt

* Tabeller, kolumner och datatyper måste ha vettiga namn och vara genomtänkta.
* En produkt ska kunna ingå i flera kategorier, en kategori ska kunna ha flera produkter. Exempelvis skulle en Playstation 4 kunna ingå i både *Skoj-* och *Elektronik*-kategorin.
* Beställningsinformation ska lagras i en beställningstabell. Minst så behövs: **Datum**, **totalpris** och information om **vilken användare** beställningen är kopplad till.
* Eftersom kunden inom en snar framtid ska införa en sökfunktion på produktnamn vill kunden att **produkt-tabellen förbereds för effektiv sökning på produktnamn**.

### Statistikfrågor

För att kunden ska bli **VG**-nöjd önskas SQL-frågor för att ta fram följande statistik:

* Total summa för alla beställningar i databasen.
* Genomsnittspris för beställningar grupperat på huruvida användarna tar emot e-postutskick eller inte. Exempelresultat:

|  |  |
| --- | --- |
| Genomsnittspris för beställningar | Tar emot e-postutskick |
| 7000 | 0 |
| 4100 | 1 |

# Betygssättning

Vid rättning kommer ett antal faktorer spela in, så några tips att tänka på:

* Finns all funktionalitet på plats? Läs igenom det här dokumentet **NOGA**.
* Är webbapplikationen stabil? Valideras inmatningar?
* Är designen någorlunda vettig? Förstår man hur sidan fungerar?
* Kodmässigt – är koden läsbar och förståelig? Är namngivningen bra?
* Har klasser använts på ett bra sätt? Hur är objektorienterings-tänket?

### Databasrelaterat:

* Fungerar scriptet?
* Är databasen vettigt designad? Namngivning, nycklar etc.
* Är kolumnernas datatyper genomtänkta?
* Är statistikfrågorna fungerande och även välskrivna?

**SLARVA INTE!**

När ni känner er klara – lämna inte in direkt utan sluta koda i någon dag. Titta sedan igen på det med nya, fräscha ögon innan ni lämnar in.

# Inlämning

Som tidigare nämnt är uppgiften **individuell** och alla måste lämna in en egen lösning. Självklart får ni dock sitta och hjälpa varandra framåt. Det rekommenderas till och med.

Packa ihop all kod och ge mig alltihop – exempelvis på USB-minne eller via en dropbox-länk. Se till att alltihop är med, jag vill kunna öppna er solution och kolla den i Visual Studio. **SQL-script** för att skapa databasen och för att lägga in testdata (lägg gärna allt i ett script) ska lämnas in tillsammans med **koden**.

Skicka inte e-post med en enda stor packad fil, sådana filer brukar blockeras av våra mailfilter.

Ingen dokumentation behövs.

Både programmeringskursen och databaskursen betygssätts utifrån den här (och förra) inlämningen.

**Deadline är onsdag 30:e mars klockan 23:59.** Ni får dock lämna in så fort ni känner er klara – se bara till att ni testar ert program noga själva först. Be gärna någon annan prova också.

Om något är oklart gällande kraven så fråga mig så reder vi ut det.

**Lycka till och ha så roligt!**