TEKNISKA HÖGSKOLAN

HÖGSKOLAN I JÖNKÖPING

UTVECKLING AV E-HANDELSSYSTEM MED IMPLEMENTERAD BETALLÖSNING

Christian Andersson

Fredrik Josefsson

Rickard Pettersson

EXAMENSARBETE 2007

DATATEKNIK

TEKNISKA HÖGSKOLAN

HÖGSKOLAN I JÖNKÖPING

UTVECKLING AV E-HANDELSSYSTEM MED IMPLEMENTERAD BETALLÖSNING

DEVELOPING AN E-COMMERCE SOLUTION WITH IMPLEMENTED PAYMENT SYSTEM

Christian Andersson

Fredrik Josefsson

Rickard Pettersson

Detta examensarbete är utfört vid Tekniska Högskolan i Jönköping inom ämnesområdet datateknik. Arbetet är ett led i den treåriga högskoleingenjörsutbildningen. Författarna svarar själva för framförda åsikter, slutsatser och resultat.

Handledare: Elisabet Olsson

Omfattning: 10 poäng (C-nivå)

Datum:

Arkiveringsnummer:

Postadress:	Besöksadress:	Telefon:
Box 1026	Gjuterigatan 5	036-10 10 00 (vx)
551 11 Jönköping		

Abstract

This thesis presents the development of an e-commerce solution for a clothing company. The assignment consisted of creating a working e-commerce solution where customers should feel secure while shopping. The shop shall be able to handle card transactions which mean that high security is needed. The design of the shop has to be well thought through as usability needs to be high.

The goals of the assignment are summarized in the following questions:

- How is an e-commerce solution developed?
- How is sufficient security achieved?
- How do payment transactions through the Internet function both technically and practically?
- How is safety created for the visitors of the e-commerce solution?

Based on the requirement specification that was accomplished, the foundation was laid down for the e-commerce solution together with ASP.NET and C#. XHTML and CSS were used to get a clean intuitive design. A database was created in SQL Server 2005 and is used to store all the information.

The result is an Internet shop with usable design, and an implemented payment solution which enables secure transactions with aid from SSL and 3D Secure. The database can be used in future purposes, as it contains advanced functions aimed for international systems.

To get an insight in how people in general apprehend payments and security on the Internet a questionnaire was made and analyzed. The project has showed that applications made in ASP.NET are well suitable to be used for e-commerce solutions.

Sammanfattning

Detta examensarbete behandlar utvecklingen av en e-handelslösning åt ett företag som handlar med kläder. Uppdraget bestod av att skapa en fungerande e-handelsplats där kunder ska känna sig trygga när de handlar. Butiken ska kunna hantera kortbetalningar så hög säkerhet krävs på sidan samtidigt som den ska vara väl genomtänkt i sin design då användbarheten ska vara hög.

Uppdragets mål har sammanfattats i följande frågeställningar:

- Hur utvecklas ett e-handelssystem?
- Hur uppnås tillräcklig säkerhet?
- Hur fungerar betaltransaktioner via Internet tekniskt och praktiskt?
- Hur skapas en trygghet för e-handelsplatsens besökare?

Baserat på kravspecifikationen som utfördes lades grunden till e-handelslösningen med hjälp av ASP.NET och C#. XHTML och CSS-mallar användes även för att få en stilren intuitiv design. En databas skapades i SQL Server 2005 för att lagra all information som sidan innehåller.

Resultatet är en Internetaffär med användbar design, och en implementerad betallösning så att säkra betalningar kan ske med hjälp av SSL och 3D Secure. Databasen kan även användas i framtida syften, då den innehåller avancerade funktioner riktade åt internationella system.

För att få en inblick i hur människor i allmänhet uppfattar betalningar och säkerhet över Internet genomfördes och analyserades en enkät med ett antal frågor. Arbetet har visat att applikationer gjorda i ASP.Net är lämpliga att nyttja för e-handelslösningar.

Nyckelord

E-handel, användbarhet, serverteknik, klientteknik, betalteknik, kryptering, betalväxel, digital plånbok, databasteori

Figurlista

Kodexempellista

Innehållsförteckning

1 Inledning

1.1 Bakgrund

De flesta företag inom säljbranschen har för länge sedan insett fördelarna med att sälja sina produkter på Internet med hjälp av ett e-handelssystem. Det finns inte många outforskade områden kvar inom e-handel, men en fråga som har för vana att dyka upp är hur ett företag tar betalt av sina kunder via kreditkort och hur dessa kunder ska känna sig trygga. Många är fortfarande oroliga över att lämna ut sitt kreditkortsnummer på Internet på grund av risken för bedrägeri. Rapporten kommer därför att i huvudsak handla om hur betaltransaktioner fungerar på Internet och vilka alternativ som finns i dagsläget och kommer i framtiden.

För att tillämpa det som beskrivs har ett nystartat företag som behöver en komplett e-handelslösning kontaktats. Företaget i fråga säljer exklusivare kläder och vill expandera sin verksamhet för att nå ut till en bredare grupp människor via Internet. En e-handelsplats ska utvecklas åt detta företag och tillvägagångssättet kommer att beskrivas i rapporten.

Under utbildningens tid på Ingenjörshögskolan i Jönköping har vi skapat många olika applikationer, däribland en Internetbutik. Den butiken var dock i mycket mindre omfattning, både i databasdesign samt i sin kodning då inga större kodningskunskaper krävdes.

Med detta examensarbete ville vi skapa en sida där både design, databas, kod samt säkerhet var av mycket hög kvalitet.

Arbetet som presenteras i denna rapport är en del av vår kandidatexamen i Datateknik vid Ingenjörshögskolan i Jönköping.

1.2 Uppdrag

Uppdragsgivaren för detta projekt är ett nystartat företag som kallar sig BRND Clothing & Accessories som har för avsikt att saluföra exklusivare kläder via Internet.

Företaget vill ha en fungerande e-handelsplats där kunder ska känna sig trygga när de handlar. Butiken kommer att ha kunder från hela Norden så kortbetalningar måste kunna hanteras. Ett administrationsgränssnitt där butiken kan administreras måste också finnas.

Eftersom butiken kommer att erbjuda kläder till försäljning så är det viktigt att kunder får så mycket information som möjligt om en produkt så att de vågar ta steget att handla. Skulle ett klädesplagg inte passa så ska det vara enkelt att via hemsidan få information om hur ett byte sker.

Målgruppen för butiken är trend- och stilmedvetna så butikens utseende ska återspegla detta samtidigt som stora krav på användbarhet ställs.

I framtiden är det möjligt att företaget önskar att expandera inom andra områden. Därför ska butiken vara anpassad för att hantera olika sorters varuslag.

1.3 Syfte och mål

Syftet med examensarbetet är att skapa en fungerande e-handelsplats åt företaget, som vill sälja kläder över Internet. Ett resultat av detta kommer förhoppningsvis att bli att företaget ökar sin försäljning och får många nöjda kunder.

Det största målet sett ur vårt perspektiv är självklart att uppfylla företagets mål, men också att få mer kunskap om objektorienterad webbutveckling och säkerhet på Internet. Säkerhet är oerhört viktigt och det tros vara en stor merit att kunna mer inom detta område.

Vi vill också utveckla en e-handelslösning som är enkel att anpassa för andra områden än kläder. Därför anses det vara viktigt att redan från början planera noggrant och ha detta i åtanke vid konstruering av exempelvis databasen.

Kunder ska känna att de vill handla i e-handelslösningen, det ska vara lätt att använda sidan och designen ska vara enkel och stilren.

1.4 Frågeställningar

De mål som har beskrivits kan sammanfattas med ett antal frågeställningar som ska besvaras i rapporten:

- *Hur utvecklas ett e-handelssystem?*
- *Hur uppnås tillräcklig säkerhet?*
- *Hur fungerar betaltransaktioner via Internet tekniskt och praktiskt?*
- *Hur skapas en trygghet för e-handelsplatsens besökare?*

1.5 Avgränsningar

Rapporten omfattar ej utvecklandet av administrationsgränssnittet.

1.6 Disposition

Teori
Beskriver och förklarar kunskapen som krävs för att förstå hur genomförandet har skett. Det som kommer behandlas är de vanligaste servertekniker, klienttekniker och programmeringsmetoder som finns för webbutveckling. Även vad användbarhet är, vilka betaltekniker som finns och hur god säkerhet uppnås kommer att beskrivas.

Genomförande
Beskriver de moment som har genomförts för att nå fram till resultatet. Det redogörs även för en enkät som handlar om betalning över Internet och säkerhet. Kravspecifikationen har även placerats i detta kapitel.

Resultat
Behandlar det resultat som har skapats genom arbetet.

Slutsats och diskussion
Redovisar författarnas egna kommentarer till hur resultatet blev.

2 Teoretisk bakgrund

2.1 Serverteknikker

2.1.1 Relationsdatabaser

För att kunna förstå vad en relationsdatabas är, måste ordet databas förklaras. En databas kan ganska lätt förklaras som en lagrad, strukturerad samling av information. Det ska även vara av relaterade slag, så t.ex. information om elever och antal poäng är en databas.

För att få en bra överblick över databaser som är stora används ofta något som kallas för relationsdatabaser. Detta är den vanligaste formen av databaser.[1] Den är uppbyggd av ett antal informationshållande tabeller som kopplas ihop med så kallande relationer. Med hjälp av relationerna kan mycket stora databaser skapas.

En databas kan innehålla många olika tabeller, en tabell kan vara för Kund, en annan för Artikel osv. Tabellerna består utav kolumner och rader, kolumnerna kallas för fält och raderna för post. Varje fält lagrar en typ av data vilken definieras av datatypen som det specifika fältet har. Posterna är helt unika och ska innehålla egenskaper och en identifierare. Identifieraren brukar ofta kallas för nyckel och den gör så att det lätt går att identifiera en viss post i tabellen.

För att visa hur relationsdatabaser fungerar, har Figur 1 skapats.

Kundnummer	E-post	Fornamn	Efternamn	Gatuadress
123	nisse@hotmail.com	Nisse	Persson	Storgatan 2B
124	maria@gmail.com	Maria	Karlsson	Storgatan 3C

Figur 1 Exempel på kundtabell

Som synes ger detta en bra överblick över vilka kunder som finns. Identifierare är Kundnummer medan E-post, Fornamn, Efternamn och Gatuadress är egenskaper.

2.1.1.1 Relationer

Svaret på varför relationsdatabaser används så ofta, är det att när de används, undviks dubbellagring. Exemplet i Figur 1 kan användas igen, samt ett nytt exempel, se Figur 2.

[1] Databasteknik, Thomas Padron-McCarthy och Tore Risch, Studentlitteratur 2005, ISBN 9144044496

Artikelnummer	Tillverkare	Namn	Pris	Kundnummer
500	Asus	P5B	900	123
501	Quiksilver	Brigade	250	123

Figur 2 Exempel på artikeltabell

Figur 2 visar en del av en artikeltabell för ett företag, som vill ha koll på vem som har köpt vad. Istället för att skriva in all information som redan finns i kundtabellen, vilket skulle vara onödigt då samma information sparas flera gånger och det skulle bli mer minneskrävande, så läggs den främmande nyckeln ”Kundnummer” in i artikeltabellen.

Denna främmande nyckel gör att informationen från artikeltabellen refererar till kundtabellen, där kunden lätt identifieras med hjälp av kundnummer. En relation mellan tabeller har uppstått som hjälper till att hålla informationsmängden nere samtidigt som det sparar minne.

Totalt finns det tre olika sorters tabellrelationer, de är:

- *1:1 – En till en*
  Det finns för varje post i en tabell en motsvarande post in den andra tabellen som ingår i relationen.

- *1:N – En till många*
  Istället för att endast ha en motsvarande post i den relaterade tabellen, innebär den här relationen att en post i en viss tabell kan ha en eller många poster i en eller fler tabeller.

- *N:N – Många till många*
  Innebär att många poster i en tabell kan vara relaterade till många andra poster i andra tabeller. När en N:N relation skapas, så läggs även en mellanliggande tabell till relationen. I denna tabell läggs primärnycklarna från de båda tabellerna, de blir då främmande nycklar.

2.1.1.2 Normalisering

Normalisering är en arbetsmetod som används för att undvika dubbellagring, även kallat redundans. De tre vanligaste formerna som alltid bör används, för att göra en minnessnål databas, är:

- *1NF – Första normalformen*
  Definitionen lyder: *”En relation är på 1NF om dess termer är odelbara och uppträder endast en gång”*

  Vilket betyder att endast ett värde får finnas i varje post i en tabell. T.ex. om en kund har många telefonnummer, då får de inte lagras i en och samma post. De måste delas upp till flera poster. [2]

- *2NF – Andra normalformen*
  Definitionen lyder: *”En relation är på 2NF om den är på 1NF och varje attribut (egenskap) beror på hela nyckeln.”*

  Ett exempel visar detta bäst:
  Deltagare: **Personnr** **Kursnr** Kursnamn Start
  Denna är inte på 2NF, för Kursnamn kan härledas genom Kursnr. Kursen ändrar inte namn för varje deltagare på kursen. För att göra om denna så den passar in på 2NF, måste en ny relation skapas. När det är gjort finns det två stycken relationer.

  Kurser: **Kursnr** Kursnamn
  Deltagare: **Kursnr** **Personnr** Start
  Egenskapen Start är nu beroende av båda nyckelattributen, start är heller inte lika för alla Kursnr och kurserna startar inte på samma dag för alla Personnr.[3]

- *3NF – Tredje Normalformen*
  Definitionen lyder: *”En relation är på 3NF om den är på 2NF och ingen egenskap är transitivt beroende av nyckelbegreppet.”*

  Transitivt betyder att ett attribut inte får identifiera ett annat attribut. Ett exempel på detta är:
  Faktura: **Fakturanr** Kundnr Kundnamn Fakturadatum
  Som synes kan Kundnamn fås fram genom både Fakturanummer och Kundnr. Detta är vad som menas transitivt beroende. Istället för att ha det så här, delas tabellen upp.

  Faktura: **Fakturanr** ↑Kundnr Fakturadatum
  Kund: **Kundnr** Kundnamn
  När symbolen ↑ läggs till framför attributet Kundnr betyder det att det är en främmande nyckel, dvs är en nyckel i en annan tabell.[4]

[2] OOS/UML, s.280, Mats Apelkrans & Carita Åbom, Studentlitteratur 2001, ISBN 9144021380
[3] OOS/UML, s.282, Mats Apelkrans & Carita Åbom, Studentlitteratur 2001, ISBN 9144021380
[4] OOS/UML, s.283, Mats Apelkrans & Carita Åbom, Studentlitteratur 2001, ISBN 9144021380

2.1.1.3 SQL

När en sökning sker i en databas används ett språk som kallas för frågespråk, och själva sökningen kallas för en ”fråga” eller en ”query”. SQL är ett standardiserat frågespråk som används till relationsdatabaser. Det utvecklades från början av IBM men under tidens gång utvecklades många olika versioner. ANSI[5] antog år 1986 den första formella specifikationen, och ett år senare, 1987, antog även ISO[6] specifikationen. Efter några år sågs standarden över för att ta fram en ny specifikation. Detta var 1992, och specifikationen kallas då även för SQL-92.

Den version som används i Microsoft SQL Server 2005 kallas för T-SQL (Transact-SQL) och det är Microsoft som har varit med och utvecklat denna SQL-dialekt. Det finns ett antal nya finesser i T-SQL och några av dem är att lokala variabler kan introduceras i ett script, och att en uppdatering i form av en FROM-sats som kan placeras inuti ett DELETE- eller UPDATE-kommando vilket innebär att de är lite mer kraftfulla.[7]

SQL kan delas in i två delar:

- *DDL – Data Definition Language*
  Detta är de satser och konstruktioner som används när en konstruktion av en ny databas sker.

- *DML – Data Manipulation Language*
  När konstruktionen av databasen är klar så används nya satser och konstruktioner för att kunna hantera informationen i databasen. Egentligen är det endast denna del av SQL som kan kallas för ett s.k. frågespråk.

Det finns många olika utvecklingar av SQL som medför att saker som fungerar i en version behöver ej fungera i en annan. Det finns ett antal kommandon som bör behandlas då dessa är vanliga inom SQL.

[5] ANSI, American National Standards Institute, grundades år 1918. Det är ett institut som definierar standarder som kan standardiseras. Den mest kända ANSI-specifikationen är ASCII-teckenuppsättningen. Officiell hemsida: http://www.ansi.org/
[6] ISO, International Organisation for Standardisation, är ett nätverk av nationella standardinstitut från 157 länder. Officiell hemsida: http://www.iso.org/
[7] http://www.devguru.com/technologies/t-sql/home.asp (Acc. 2006-11-26)

- *SELECT – Välja ut data*
  Det är det mest använda kommandot inom SQL. Används när en speciell tabell, kolumn eller rad ska hämtas. Det går även att välja hur informationen ska presenteras, om den ska vara sorterad på ett särskilt sätt.

  Ett exempel på en SQL-sats med SELECT:
  SELECT Efternamn FROM Kund ORDER BY Efternamn
  Satsen väljer ut alla efternamn från tabellen Kund och sorterar dem i bokstavsordning. [8]

- *INSERT – Lägga till data*
  När detta kommando används läggs en ny post till i en vald tabell. Kommandot skrivs som INSERT INTO som efterföljs med VALUES. Det är den data som ska skrivas in i tabellen.

  Ett exempel på en SQL-sats med INSERT:
  INSERT INTO Kund (Fornamn, Efternamn, Personnummer, E-post)
  VALUES (’Nisse’, ’Svensson’, ’800115-1234’, ’Nisse@gmail.com’)
  Denna sats skapar en ny post i tabellen Kund där Nisse skrivs in i Fornamn, Svensson i Efternamn osv.[9]

- *UPDATE – Ändra data*
  Används när data ska uppdateras. INSERT och UPDATE kan kännas lika varandra, men UPDATE lägger inte till en ny post, det modifierar endast en gammal post. Kommandot skrivs som UPDATE följt av tabellen som ska ändras, SET för att definiera vad som ska ändras samt den nya datan. Till sist kommer WHERE, som definierar vilken post som ska ändras.

  Ett exempel på en SQL-sats med UPDATE:
  UPDATE Kund SET E-post =’Kalle@gmail.com’ WHERE Kundnr= ’323’
  Satsen ändrar e-posten till Kalle@gmail.com för kunden med kundnr 323.[10]

[8] Lär dig SQL på 3 veckor, s. 20 , Ryan K Stephens, Ronald R. Plew, Bryan Morgan, Jeff Perkins, Pagina Förlags AB 1997, Göteborg ISBN 9163604973
[9] Lär dig SQL på 3 veckor, s. 154 , Ryan K Stephens, Ronald R. Plew, Bryan Morgan, Jeff Perkins, Pagina Förlags AB 1997, Göteborg ISBN 9163604973
[10] Lär dig SQL på 3 veckor, s. 161 , Ryan K Stephens, Ronald R. Plew, Bryan Morgan, Jeff Perkins, Pagina Förlags AB 1997, Göteborg ISBN 9163604973

- *DELETE – Radera data*
  Tar permanent bort data från en databas. Det går att välja om endast en rad, flera rader eller alla rader ska tas bort från tabeller. Skrivs som DELETE FROM samt vilken tabell som ska ändras i, och till sist WHERE för att definiera vad som ska tas bort.

  Ett exempel på en SQL-sats med DELETE:
  DELETE FROM Kund WHERE Kundnr='323'
  Satsen tar bort kunden med kundnr 323.[11]

- *JOIN – Hämta data från två eller fler tabeller*
  När ett resultat ska visas så kan det kräva att data hämtas från två eller fler tabeller. Då måste en JOIN användas. Det går även att välja data med en JOIN på olika sätt.

  INNER JOIN listar alla rader från båda tabeller där det finns en träff. Om det finns rader i tabell 1 som inte finns i tabell 2, så listas inte de raderna.

  LEFT JOIN listar alla rader från tabell 1, även om det inte finns några träffar i tabell 2. Om det finns rader i tabell 1 som inte finns i tabell 2, så listas även dessa rader.

  RIGHT JOIN listar alla rader från tabell 2, även om det inte finns några träffar i tabell 1. Om det finns rader i tabell 2 som inte finns i tabell 1, så listas även dessa rader.[12]

2.1.1.4 Databashanterare

Program som lagrar data samt hanterar databaser kallas för databashanterare eller DBMS (Database Management System). De mest kända är MS Access, MySQL, Oracle samt MS SQL Server. Microsoft SQL Server är en kraftfull, enkel och flexibel databashanterare, som är vanlig när det handlar om affärer på Internet.[13] SQL Server körs enbart på webbservern, så det körs aldrig lokalt på en användares dator. Det är även designat för att klara av ett stort antal användare samtidigt, där gränsen är det totala minnet som servern har, i motsats till MS Access där gränsen redan nås vid 255 användare.[14]

[11] Lär dig SQL på 3 veckor, s. 164 , Ryan K Stephens, Ronald R. Plew, Bryan Morgan, Jeff Perkins, Pagina Förlags AB 1997, Göteborg ISBN 9163604973
[12] http://www.w3schools.com/sql/sql_join.asp (Acc. 2006-12-03)
[13] http://www.microsoft.com/sql/default.mspx (Acc. 2006-11-26)
[14] http://www.mssqlcity.com/Articles/Compare/sql_server_vs_access.htm (Acc. 2006-11-26)

Det finns även andra alternativ som kan användas när en databas ska byggas. Figur 3 visar en jämförelse mellan de mest kända databashanterare och för att inte försvenska alla ord, kommer jämförelsen inte att översättas.[15]

	MS SQL 2005 Express	Oracle Database XE	MySQL 5.0
Numbers of Processors	1	1	Limited only by the OS
Max Database Size	4 GB	4 GB	65+ GB per table
Max RAM	1 GB	1 GB	Limited only by the OS
OS Availability	Windows	Windows; Linux	Windows; Linux; BSD; Netware; others
Upgradeable	Yes	No	No path available
Included GUI Management Tool	No, available separately	Yes; Web based	No; Third party available
64-Bit Support	Yes	No	Yes
Support for Stored Procedures	Yes	Yes	Yes
Support for Views	Yes	Yes	Yes
Support for Triggers	Yes	Yes	Yes
Support for Replication	Yes	Yes; Undocumented	Yes
Support for XML	Yes	No	Yes
Auto Tuning	Yes	No	No
Automated Scheduling	No	Yes	OS Support (cron, scheduled tasks, etc.)
Reporting Services	Yes (SQL Report Server)	Yes (Using HTML DB)	Third party
Technical Support Available	Yes ($245 call; $99 online)	No	Yes (from $595 to $4,995 per year per server)

Figur 3 Jämförelse mellan databashanterare

2.1.1.5 Lagrade procedurer

Det är ett program eller procedur som är fysiskt sparad i en databas, vilket skiljer mot SQL injections, vilket är vanliga SQL-kommandon som är skrivna i en sidas källkod. Lagrade procedurer är ofta, men inte alltid, använd för att utföra SQL-kommandon i en databas. De flesta databassystem som stödjer lagrade procedurer gör detta med sin egen version av SQL. Microsoft SQL Server använder T-SQL.

Det finns många fördelar med lagrade procedurer i jämförelse med vanliga SQL-kommandon som skrivs i sidans kod, bl.a.:

- *Förkompilering av frågor*
  Frågor sparade som en procedur kan köras fortare många gånger, eftersom de kan bli förkompilerade med informationen från frågan. Detta gör att det minskar minnesmängden jämfört med vanliga SQL. Det kan även skapa ett problem om inte all information finns när förkompileringen exekveras, då tar det faktiskt längre tid än vanlig SQL.

[15] http://downloads.techrepublic.com.com/5138-9592-6028761.html (Acc. 2006-11-26)

- *Exekvering på en specialiserad server*
  Lagrade procedurer körs direkt mot databasen. Detta gör att i ett produktionssystem kan procedurerna köras på en specialiserad server, som har direkt åtkomst till den tillgängliga informationen, vilket i sin tur gör att frågor kan exekveras snabbare.

- *Plats av exekvering*
  När en användare sitter hemma och jobbar mot en server, så skickas vanligtvis SQL-frågor en åt gången. Detta gör att om många frågor behövs, kan det ta lång tid innan resultatet från alla frågor kan skickas tillbaks till användaren till slut. Då en procedur kan innehålla många olika komplexa SQL-kommandon, vilka kan köras alla på en gång, så sparas både nätverkstrafik och tid. Databasservern behöver endast skicka tillbaka ett resultat, inte många olika.

- *Säkerhet*
  Bra skrivna procedurer kan vara väldigt säkra. Då det även är möjligt att kryptera lagrade procedurer möjliggör detta för väldigt hög säkerhet. Det går att attackera vanliga SQL-kommandon med något som kallas för SQL-injections attacker. Hur detta går till är att en användare skriver in egen SQL-kod på en sida. Det skulle t.ex. kunna vara vid en inloggning, då är det möjligt att hacka sig in på sidan, eller in i databasen. Om lagrade procedurer används, kan en användare inte hacka sig in på detta vis.[16]

2.1.2 Microsoft .NET

Microsofts definition på .Net:
”The .NET Framework is a development and execution environment that allows different programming languages & libraries to work together seamlessly to create Windows-based applications that are easier to build, manage, deploy, and integrate with other networked systems.”[17]

Microsoft .NET, vilket ofta skrivs Dotnet, är en uppsättning datorprogram som sammankopplar system, enheter och information.

[16] http://builder.com.com/5100-6388-5083541-2.html (Acc. 2006-11-26)
[17] http://msdn.microsoft.com/netframework/gettingstarted/default.aspx (Acc. 2006-09-24)

- *.NET Framework*
  Detta är en plattform som är standardiserad, som används för att köra .NET-program. Det går att använda vilket språk som helst, det som krävs är att programmeraren har en kompilator för språket. Kompilatorn översätter koden till Microsoft Intermediate Language, MSIL. Microsoft Intermediate Language är en s.k. byteskod som kompileras av en JIT-kompilator (Just-In-Time) när programmet körs av .NET Framework. Detta betyder att en användare måste ha samma version av .Net framework installerat på sin egna dator.

- *CTS, Common Type System*
  Den standardiserade del som har hand om olika typer som möjliggör att program och komponenter kan vara språkoberoende.

- *CLR, Common Language Runtime*
  Detta kan kallas för kärnan i .NET Framework, det används för att köra programmen. Det tar hand om många olika saker, bl.a. ett objekts livscykelhantering, kodsäkerhet samt profilering.

- *CLI, Common Language Infrastructure*
  Är den standard, ECMA-335[18] samt ISO 23271[19], som .Net Framework följer. Det beskriver hur program ska kunna köras i ett flertal olika miljöer utan att behöva skrivas om. Standarden innehåller bl.a.:

  Filformat
  CTS, Common Type System
  MSIL, Microsoft Intermediate Language
  Utbyggbart metadatasystem
  Klassbibliotek

- *CLS, Common Language Specification*
  Den standard som alla .NETspråk ska uppfylla för att kunna köras i framework. Några av de språk som stöds är: [20]

  C#
  C++
  Visual Basic.NET
  J#
  Python
  Perl
  COBOL

[18] En internationell standard som definierar CLI så att program kan köras på multipla system utan att skrivas om. Standarden består av sex delar som i sin tur består av egna instruktioner.
[19] En internationell standard som definierar CLI så att program kan köras på multipla system utan att skrivas om. Standarden består av fem delar som i sin tur består av egna instruktioner.
[20] http://msdn.microsoft.com/netframework/technologyinfo/overview/default.aspx (Acc. 2006-09-24)

2.1.2.1 ADO.NET

Det är ADO.NET som ger databasåtkomst vid användning av .NET. Det är en vidareutveckling av den äldre tekniken som användes för att koppla applikationer till databaser, som kallades för ADO. ADO står för ActiveX Data Objects. Det finns ett antal skillnader mellan den gamla och den nya versionen, bl.a.:

- ADO använder en koppling mot databasen hela tiden, då allt sker i realtid. ADO.NET däremot kopplar till databasen och gör en XML-kopia på informationen för att sedan koppla ner. Vilket betyder att den nya versionen är väl lämpad för webbapplikationer. [21]

- ADO använder OLE DB för att skicka och ta emot data. ADO.NET använder en s.k. data adapter som OleDbDataAdapter, SqlDataAdapter, OdbcDataAdapter eller OracleDataAdapter. Detta gör att det går att välja hur data ska skickas till databasen, allt för att få en optimerad databas. Detta skulle kunna ske genom prestandaoptimering, genom datakontroller eller lägga till extra funktioner. [22]

2.1.2.2 ASP.NET 2.0

Microsoft har en egen plattform för webbprogrammering som kallas för ASP.NET. Den används för att skapa dynamiska och interaktiva webbsidor. ASP.NET är baserat på .NET framework. Det finns vissa tekniker som använder enklare skriptspråk, men ASP.NET använder ett riktigt programspråk. Det har även ett stort klassbibliotek som gör det till ett kraftfullt val för utveckling. ASP.NET är serverbaserat så all kod exekveras på servern för att sedan skicka den önskade sidan till klienten. Det finns komponenter som känner av vilken webbläsare en klient använder, så att sidan visas precis så som det var tänkt. Den nyaste versionen heter ASP.NET 2.0.

En av de största fördelarna med att använda ASP.NET är att det finns en separation av kod (code-behind) och design, vilket möjliggör att designer och programmerare kan arbeta parallellt. All kod som finns i code-behind läggs i separata filer. När koden sedan kompileras skapas dll-filer vilket gör att koden blir omöjlig att läsa för utomstående samt att koden läses fortare av applikationen.

[21] http://www.informit.com/articles/article.asp?p=31098&rl=1 (Acc. 2006-09-24)
[22] http://msdn.microsoft.com/library/default.asp?url=/library/enus/vbcon/html/vbconadopreviousversionsofado.asp (Acc. 2006-09-24)

2.2 Klienttekniker

2.2.1 W3C

W3C, World Wide Web Consortium, är ett internationellt konsortium med närmare 450 organisationer som medlemmar. Det skapades för att:

- *"To lead the World Wide Web to its full potential by developing protocols and guidelines that ensure long-term growth for the Web."* [23]

Att leda webben till sin fulla potential uppnås genom att utveckla gemensamma tekniska protokoll och standarder som stärker webbens utveckling och säkrar dess interoperabilitet. Bland de tjänster som W3C erbjuder finns bl.a. en databas med information om Internet vilken är riktad till utvecklare och användare. Det finns även prototyper och exempel på användningen av ny teknologi. För att säkerställa att hemsidor är kompatibla för alla webbläsare erbjuder W3C valideringsfunktioner.

De tekniska standarder som W3C har utvecklat är bl.a. HTML, CSS, XML, WCAG[24] och SVG[25]. Om en teknik används som inte är en W3C-standard, t.ex. Flash eller PDF bör den informationen finnas i något annat standardformat på sidan.

2.2.2 HTML och XHTML

- *HTML, Hyper Text Markup Language*
  Är ett programspråk som skapar webbsidor. Det är ett relativt enkelt språk som byggs upp av taggar, vilka beskriver var saker och ting ska visas på en sida. En tagg är alias eller ord som står inom två tecken, "<" och ">", därav namnet tagg. Kodexempel 1 visar hur HTML skrivs.

```
<html>
<body>

<!--Här skrivs HTML-koden--!>

</body>
</html>
```

Kodexempel 1 Hur HTML skrivs

[23] http://www.w3.org/Consortium/ (Acc. 2006-11-26)
[24] WCAG (Web Content Accesibility Guidelines) är en W3C-standard vars innehåll riktar sig till utvecklare inom Internet. Ett antal riktlinjer finns utformade som talar om hur webbsidor ska vara utformade för att vara lättillgängliga. Riktlinjerna finns på:
http://www.sics.se/w3c/resources/office/translations/wcag_full-checklist_se.html
[25] SVG (Scalable Vector Graphics) är en xml-baserad standard för skalbar vektorgrafik.

För att en sida ska fungera, ska html taggen alltid stå först, men även sist med ett ”/” i taggen. Det tecknet visar att det är en sluttagg, så att webbläsaren vet var koden och sidan slutar. Taggar måste avslutas i rätt ordning, annars kan det bli oordning på sidan.

- *XML, eXtensible Markup Language*
  Ett beskrivningsspråk som påminner om HTML. Det är till för att beskriva data, men inte för att publicera data som HTML är.

- *XHTML, eXtensible Hypertext Markup Language*
  Detta är uppföljaren till HTML, som baseras på XML. Detta gör att XHTML är en renare och mer strikt version av HTML då alla alias måste skrivas med gemener, alltid använda sluttaggar, samt använda citationstecken eller apostrofer runt attributvärden. Precis som HTML kan XHTML använda sig av CSS-mallar och JavaScript.[26]

2.2.3 CSS

CSS står för Cascading Style Sheets, de används för att göra stilmallar som skall användas på webbsidor. Det finns ingen gräns för hur många sidor som kan styras av en CSS-mall, då en länkning till mallen lätt kan skrivas in. Oftast läggs mallen som en separat fil, då de lätt kan bli väldigt stora, men det är även möjligt att lägga in definitionerna direkt i sidans kod.

Kodexempel 2 nedanför visar hur en mindre CSS-mall kan se ut.

```
body
{
    margin-top:50px;
}

h2
{
    padding-top:10px;
    margin-top:0px;
    float:right;

    font-family:'Trebuchet MS', Arial, Helvetica;
    font-size:14px;
    color:#999;
    text-transform:uppercase;
}
```

Kodexempel 2 CSS-mall

[26] http://www.w3.org/MarkUp/2004/xhtml-faq (Acc. 2006-09-24)

2.2.4 JavaScript

JavaScript är namnet på Netscape Communications Corporations implementation av ECMAScript-262. Det är ett så kallande interpreterande skriptspråk som bäddas in i en sidas HTML-kod, vilket innebär att det är webbläsaren som tyder koden.[27] Koden lämnas även orörd då den inte kompileras till maskinkod. JavaScript används ofta på webbsidor, men används även för att få tillgång till inbäddade objekt i andra applikationer.

Vanligtvis används JavaScript för att göra rörliga menyer eller beräkningar av data på en webbsida. JavaScript körs inte på servern utan det är klienten som kör det. Det går även att lägga all kod antingen i sidans HTML-kod eller lägga den i en separat fil som länkas till genom HTML-koden.

2.3 Programmeringsmetoder

2.3.1 Flerlagersarkitektur

Många gånger när en applikation ska tas fram, så delas funktionaliteten upp i separata komponenter baserat på vad de gör. Detta kallas för en n-tier arkitektur. Den allra vanligaste n-tier arkitekturen består av tre delar och kallas för tre-tier. Orsaken till att den blivit så populär är att den löser många av problemen som annars kan uppstå. Tre-tier arkitekturen består av tre delar[28]:

- *Presentation tier (användargränssnitt)*
  Kallas även för presentationslager. Ansvarar för presentation, inmatning och viss felhantering.

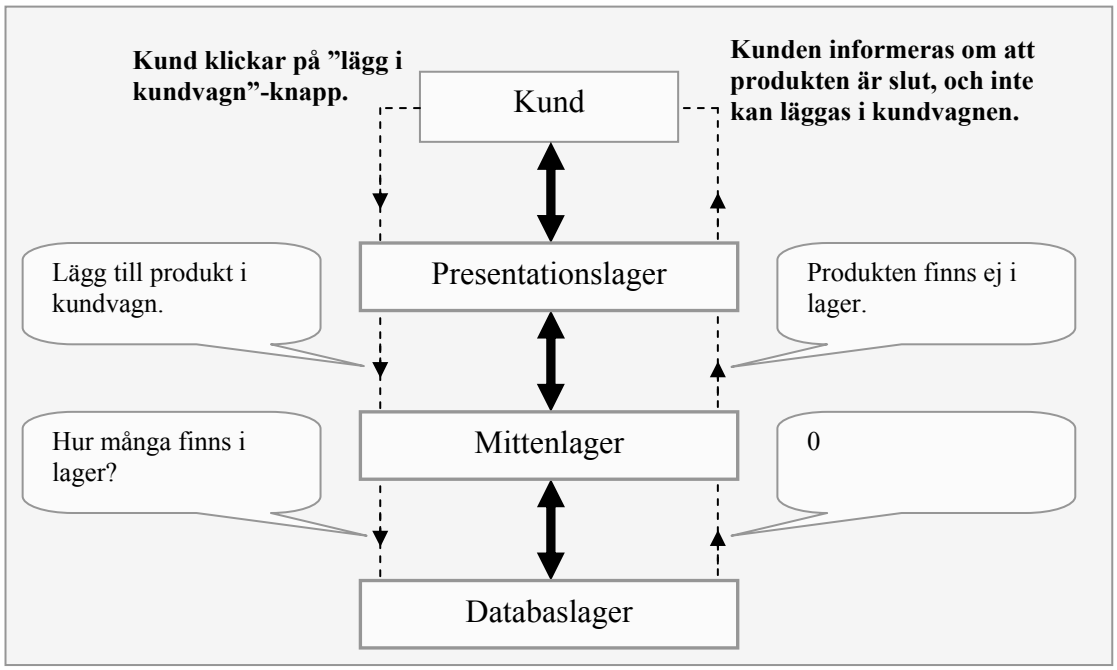
- *Business tier (mittenlager)*
  Ibland även kallat för affärslogik. Det ansvarar för beräkningar, hantering av data och kommunikation med databasen. När presentationslagret skickar en förfrågan, så är det alltid affärslogikslagret som svarar. T.ex. om en användare söker efter en vara, skickas en förfrågan från presentationslagret till mittenlagret, och oftast måste mittenlagret i sin tur skicka vidare en förfrågan till databaslagret

- *Data tier (databas)*
  Ibland även kallat för databaslager. Ansvarar för lagring av applikationens data och att skicka data till mittenlagret när det förfrågas.

[27] http://www.ecma-international.org/publications/standards/Ecma-262.htm (Acc. 2006-09-22)
[28] Cristian Darie, Karli Watson: Beginning ASP.NET 2.0 E-Commerce in C# 2005: From Novice to Professional. Apres. Sid. 13

Dessa lager är helt logiska, det finns inga begränsningar på den fysiska platsen för vardera lager. Så det går att placera varje lager på samma server, men även placera varje lager på varsin server. Det går även att dela upp ett lager i mindre bitar och placera dem på varsin server. Det enda som drar ner på möjligheterna är applikationens prestanda, då varje förfrågan måste gå genom fler servrar.

En av de viktigaste gränserna för tre-tier arkitekturer är att information måste gå i en viss order mellan lagren. Då en implementation av tre-tier arkitektur ska ske måste det ske genom att följa alla regler som finns, annars uppstår problem och det är då omöjligt att få alla fördelar som finns. Figur 4 visar hur dessa lager samarbetar.



Figur 4 Exempel på hur en kund interagerar med en tre-tier applikation [29]

2.3.2 Objektorientering

Objektorienterad programmering ,OOP, fungerar så att ett program är uppbyggt av en mängd olika objekt som interagerar. Denna metod är väldigt effektiv och kraftfull när större program konstrueras, detta eftersom programmets olika delar och dess påverkan kan minimeras. Det är även lättare att återanvända delar av ett gammalt program, då många begrepp och objektklasser ofta är generella.

[29] Cristian Darie, Karli Watson: Beginning ASP.NET 2.0 E-Commerce in C# 2005: From Novice to Professional. Apres. Sid. 16

Det finns fyra ord som måste förklaras vid OOP:

- *Klass*
  Ett program består av delar och begrepp som kallas för klasser. En klass är en teoretisk konstruktion av ett begrepp som innehåller information och funktionalitet. Varje objekt som används i ett program tillhör en särskild klass.

- *Inkapsling*
  Då programmet ska ändra något, så sker detta genom programkod som tillhör en viss klass. Så ett objekt som finns i en viss klass kan inte ändra något som finns i en annan klass, förutom då programkoden säger det.

- *Arv*
  Klasser och begrepp som används är inte oberoende av varandra, de kan ses som en sammanhängande kedja. Vissa av begreppen är generella medan andra är specialfall. Funktioner som ska användas av alla generella begrepp finns i en överklass, och specialfall kan läggas i en underklass som får andra egenskaper. En underklass ärver funktionalitet och egenskaper från den överklass som den tillhör.

- *Polymorfism*
  Det kan finnas funktioner som har yttre likheter mellan en grupp av underklasser, men som i själva verket har programmerats helt annorlunda. Gränssnittet för användaren kan se likadant ut, och det kan definieras i överklassen, men koden som exekveras finns i respektive underklass. Objekt som använder andra objekt bryr sig inte om vilket specialfall som kan tänkas använda. Utan det använder det gemensamma gränssnitt som finns definierat av överklassen.

2.3.3 Kodningsstandarder

En kod bör vara lätt att förstå, då den med stor chans kommer att underhållas av fler personer än upphovsmannen. För att koden inte ska bli svårläst är det viktigt att samma standard följs av alla för hur koden formateras. Microsoft har skapat en standard[30] som alla bör följa, men som även är för stor att behandla i denna rapport. Kort förklarat bör t.ex en knapp ha prefixet btn så att alla utvecklare vet att det är en knapp.

[30] För full insyn i standarden finns den på Internet: http://msdn.microsoft.com/library/default.asp?url=/library/en-us/cpgenref/html/cpconnamingguidelines.asp

2.4 Användbarhet

Alla Internetbutiker som finns på Internet är inte lätta att använda, många gånger blir det något fel, t.ex. vid kundregistrering eller när kunden ska slutföra sitt köp. Även om mycket tid läggs ner på att få, i detta fall, en användbar Internetbutik så kan vissa designval vara svåra att förstå.

Det behöver inte enbart vara designen det är fel på, det kan även vara programmeringsproblem eller helt enkelt dålig feedback av systemet. Kunden kan helt missa vissa funktioner som finns om de är dåligt placerade på sidan. Användbarhet är alltid ett intressant ämne och det är något som människor kommer i kontakt med varenda dag.

En annan viktig aspekt är att om en Internetbutik är svårnavigerad och svår att förstå, så gör det att kunden inte orkar använda butiken. Det kan betyda att kunden helt enkelt handlar från en annan butik, bara enbart för att användbarheten är bättre där. Så ju bättre användbarhet en Internetbutik har, så borde, teoretiskt sett, desto fler kunder använda butiken.

2.4.1 Användbarhetens definition

Definitionen för användbarhet gällande systemsammanhang finns som en internationell standard ISO 9241-11.

- *”Den grad i vilken specifika användare kan använda en produkt för att uppnå ett specifikt mål på ett ändamålsenligt, effektivt och för användaren tillfredsställande sätt i ett givet sammanhang”* [31]

Och för att förstå vad detta egentligen betyder finns även definitioner för:

- *Ändamålsenlighet*
  ”noggrannhet och fullständighet med vilken användarna uppnår givna mål.”

- *Effektivitet*
  ”resursåtgång i förhållande till den noggrannhet och fullständighet med vilken användarna uppnår givna mål.”

- *Tillfredsställelse*
  ”frånvaro av obehag samt positiva attityder vid användningen av en produkt.”

[31] http://www.usabilitypartners.se/usability/standardssv.shtml (Acc. 2006-09-22)

- *Användningssammanhanget*
  ”användare, uppgifter, utrustning (maskinvara, programvara och annan materiel) samt fysisk och social omgivning i vilken produkten används.”[32]

Den internationella standarden underlättar för all sorts utvecklare som utvecklar en produkt. Vid tester utav produkten så mäts hur många gånger ett fel görs, om det är många som fastnar vid en specifik funktion. Men även om ett fel görs så mäts den tid som det tar för testaren att återhämta sig.

Dessa tester sker som iterationer, så det går att mäta om produkten har blivit lättare eller svårare att använda. Men beroende på vad som testas, så kan olika mätningstekniker användas. T.ex. om sidans funktioner testas, så ger det ett annat resultat än om designen testas. Även om det är helt olika saker som mäts, går det att använda samma definition.

Även Jacob Nielsen[33] har utformat en definition för användbarhet.

- *Lätt att lära*
  Användaren ska snabbt kunna komma igång med sitt arbete.

- *Effektivt att använda*
  När användaren har lärt sig systemet måste det vara effektivt att arbeta med, det ska möjliggöra en hög produktivitet.

- *Lätt att komma ihåg*
  Har användaren inte använt systemet på en längre tid ska han ändå komma ihåg hur det fungerar.

- *Få fel*
  Användaren ska kunna göra så få fel som möjligt. Om han ändå gör fel ska det lätt gå att komma tillbaka till situationen där felet uppstod.

- *Subjektivt tilltalande*
  Användaren ska tycka om att jobba med systemet.[34]

[32] Användarcentrerad systemdesign, Jan Gulliksen och Bengt Göransson, Studentlitteratur 2002
[33] Jacob Nielsen är en av de mest kända användbarhetsexperterna som finns. Han har blivit kallad ’Kungen av användbarhet’ och ’Världsledande expert på användarvänlig design’. Officiell hemsida: http://www.useit.com
[34] Nielsen Jacob: Usability Engineering, Kapitel 2: What is Usability. Academic Press, Inc. San Diego, CA. 1992

2.4.2 Användbarhetsprinciper

Det finns ett antal olika riktlinjer för att uppnå en hög grad av användbarhet, de mest kända är Nielsens användbarhetsregler, Normans sju principer samt Shneidermans åtta gyllene regler. De är alla tre relativt lika varandra, då grundtanken med principerna vill uppnå samma mål.

2.4.2.1 Nielsens användbarhetsregler

I början av 1990-talet utvecklade Nielsen användbarhetsprinciper som han kallar för heuristiker. Vad de var tänkta för från början var att de skulle användas vid utvärderingar och tester, men brukar även, med fördel, användas i designfasen.

- *Systemets status ska presenteras*
  Systemet ska hålla användaren informerad om vad som pågår genom lämplig återkoppling inom lämplig tid. Användaren ska aldrig vara osäker på systemets tillstånd.

- *Det ska finnas en naturlig koppling mellan systemet och ”den verkliga världen”*
  Dialoger bör uttryckas tydligt i ord eller meningar som användaren förstår och inte i svårförståeliga termer.

- *Användarkontroll och användarfrihet*
  Användare bör snabbt kunna avsluta vissa delar av ett system, särskilt om de har hamnat fel.

- *Konsistens*
  Språk och struktur i systemet bör användas på ett och samma sätt över hela systemet.

- *Förhindra fel*
  Designa systemet så att tänkbara problem förebyggs tidigt.

- *Minimera minnesbelastningen*
  Användaren ska inte behöva minnas information från en del av dialogen till en annan. Alla funktioner i systemet ska vara väl synliga och lätta att nå.

- *Flexibilitet och effektiv användning*
  Genvägar bör finnas som kan skynda på handlingar för vana användare. Dessa bör kunna döljas om så behövs.

- *Enkel och naturlig dialog*
  En dialog bör vara minimal, naturlig och ej visa onödig information.

- *Erbjuda bra felmeddelanden*
  Meddelanden ska vara enkla och tydligt visa problemet. De bör även visa en lösning så systemet kan återhämta sig.

- *Hjälp och dokumentation*
  Dokumentationen ska vara kort och koncis, lätt att söka i och fokusera på uppgiften. Hjälptexten ska vara konkret.[35]

2.4.2.2 Normans sju principer

Normans sju principer används för att göra om krångliga uppgifter till enkla. De är framtagna för användning vid designfasen men går även att använda mellan interaktionen mellan användare och system. De kan även omformuleras lätt för att kunna användas vid utvärderingar.

- *Använd kunskap i världen och i huvudet*
  Kunskapen som krävs för att lösa en uppgift bör finnas tillgänglig i världen/systemet, en användare ska inte behöva minnas allt. Det ska även vara möjligt för en användare att förstå systemet enbart genom att interagera med det.

- *Gör uppgiftens struktur enkel*
  Uppgifter som kan utföras i systemet bör vara enkla och bör inte kräva stor planering för att kunna slutföras.

- *Gör saker synliga*
  Det ska vara tydligt att se vad som kan göras med systemet. Det ska även vara möjligt att se vad alla funktioner gör och vilka effekter de har. Om en användare gör något litet så ska även effekten vara något litet.

- *Rätt förhållanden*
  Designern ska försäkra sig om att användaren inser förhållandet mellan intention och dess effekt på systemet.

- *Utnyttja kända restriktioner*
  Naturliga restriktioner ska användas för systemet. Designa systemet på ett naturligt sätt så att endast en viss handling är tänkbar på ett särskilt sätt. Detta för att användaren inte ska kunna tänka att det kanske är ett misstag eller ett fel på systemet.

[35] http://www.useit.com/papers/heuristic/heuristic_list.html (Acc. 2006-09-22)

- *Designa för fel*
  När ett system skapas så bör man utgå ifrån att ett fel kan uppträda, och därför förbereda för detta. Det bör även vara lätt att ångra en handling. Om det inte går att ångra så vågar användaren inte utforska systemet. Det går även designa in funktioner så att användaren måste göra en viss sak. Detta skulle kunna vara att användaren måste skriva in sin e-postadress för att kunna gå vidare i registreringen.

- *När allt annat misslyckas, standardisera*
  Om det inte finns en synbar lösning på ett problem bör designern välja bästa möjligheten och använda den över hela systemet.[36]

2.4.2.3 Shneidermans åtta gyllene regler

Shneiderman har liknande regler som Nielsen, men här läggs mer fokus på användare-system interaktionen. Dessa regler kan användas både inom designfasen och vid tester av systemet.

- *Sträva efter konsekvens*
  Teckensnitt, färger, symboler, placering med mera skall följa en viss layout, så att användaren känner igen sig i systemet.

- *Genvägar för vana användare*
  Genom att erbjuda genvägar för en användare möjliggör det att förkorta de steg som måste utföras för en viss funktion.

- *Erbjud informativ feedback*
  Det är viktigt att användaren ser att det han/hon gör, faktiskt åstadkommer någonting. Systemet ska visa vad som händer, men om det handlar om mindre eller vanliga handlingar som utförs ofta så ska det vara måttlig feedback.

- *Designa dialoger som främjar avslut*
  När en sammansatt händelse sker så ska den vara organiserad så att det finns en början, mellandel och ett slut. En informativ bekräftelse ska kunna ges på att uppgiften faktiskt är avslutad.

- *Erbjud enkel felhantering*
  Designa systemet så att det inte går att göra några allvarliga fel. Om ett fel ändå sker så ska systemet upptäcka felet och föreslå enkla men konstruktiva instruktioner för att kunna lösa problemet.

- *Möjliggör att användaren kan ångra handlingar*
  Om en användare vet att om det går att ångra sin handling så blir han/hon inte lika spänd när systemet används.

[36] Norman, D. A. *The Design of Everyday Things*. London: MIT Press. 1988

- *Ge användaren initiativet över systemet*
  Om en användare inte känner att han/hon har kontroll över systemet så kan lätt känslor av missnöjdhet och ångest uppstå.

- *Reducera belastningen på korttidsminnet*
  Då det finns en begränsning i det mänskliga korttidsminnet ska det som visas på skärmen hållas så enkelt som möjlig.[37]

Ben Shneiderman säger dock själv att:
”These rules are far from complete and sometimes in conflict, but they have served as a useful starting point for design critiques.”[38]

samt:
“However, guidelines, models, and principles alone will never guarantee success. Designers have to develop their own style and then test, test, test, and test again.”[39]

Citaten visar att även om en utvecklare eller designer följer dessa regler, så behöver inte detta vara det optimala. Tester utav systemet är det som faktiskt visar ifall det fungerar eller ej.

2.5 Betaltekniker

När en kund köper varor från en Internetbutik används troligtvis någon sorts betaltjänst. Betaltjänsterna som används fungerar på olika sätt och har olika säkerhetslösningar. En del skyddar kundens kontouppgifter, eller olika köpmönster som kunden har. En del har även funktionen att garantera säljaren betalningen. När en betaltjänst ska integreras in i en Internetbutik måste vissa säkerhetsaspekter finnas med.

De största utmaningarna för e-handelssystem är:

- *Frihet att välja e-handelsmetod*
  Alltså ska systemet stödja flera betalningsmetoder, så att kunden kan välja det den föredrar.

- *Säkerhet*
  Hur säljaren gör betalningen säkrare, så det verkligen går att erbjuda säkerhet vid kundens transaktioner över Internet.

[37] Shneiderman, B. *Designing the user interface. Strategies for effective human-computer interaction*. Addison-Wesley 1998
[38] http://www.cs.umd.edu/%7Eben/Fun-p48-shneiderman.pdf (Acc. 2006-09-22)
[39] http://www.cs.umd.edu/%7Eben/Fun-p48-shneiderman.pdf (Acc. 2006-09-22)

- *Privatliv*
  Hur säljaren skyddar kunders privata information.

- *Anonymitet*
  Hur betalningen görs anonymt.

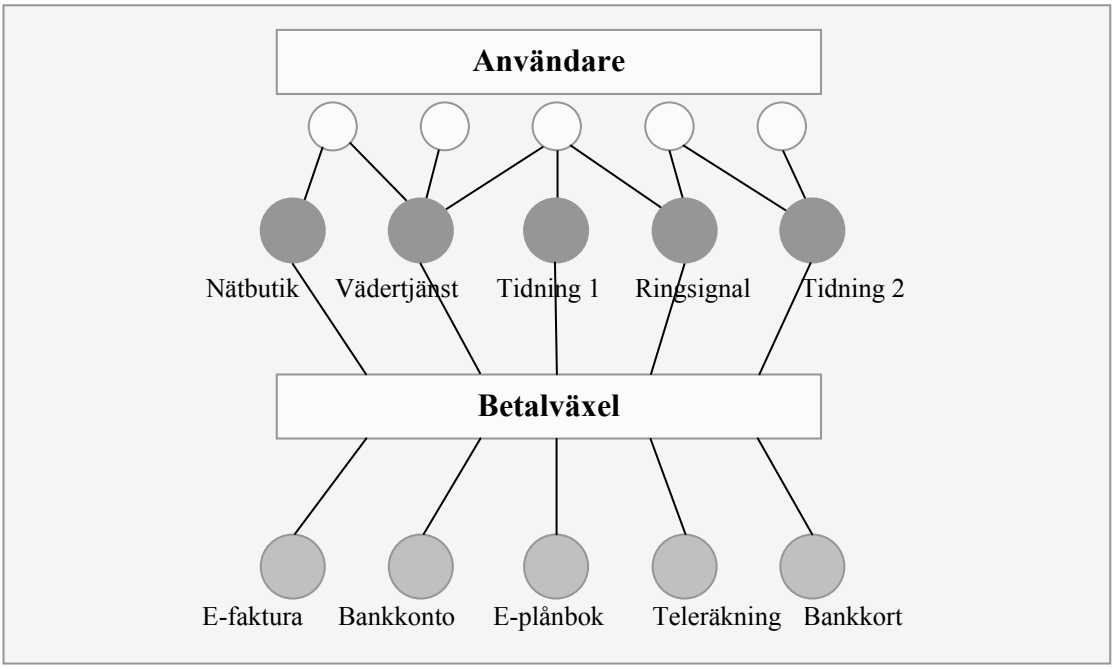
- *Risk*
  Hur säljaren minskar kundens risk vid betalningen.

- *Bekvämlighet*
  Hur säljaren erbjuder kunderna bekvämlighet.

- *Kostnad*
  Hur säljaren minskar implementations- och driftskostnader för e-handelssystemet[40]

2.5.1 Betalväxel

En betalväxel, även känt som Payment Service Provider (PSP), är ett företag som förmedlar betaltjänster men även i vissa fall betalningar mellan betaltjänsteaktörer och webbutiker. Den vanligaste betaltjänst en betalväxel erbjuder är olika kontokort och bankers direktbetalning, men det finns även många andra betaltjänster tillhängliga. Figur 5 visar hur en betalväxel förmedlar betalningar till olika tjänster.



Figur 5 En användare kan betala olika företag genom en betalväxel

[40] Payment technologies for E-commerce, *Weidong Kou*

Det går att se en betalväxel som ett gränssnitt, då Internetbutiken bara integrerar sin applikation mot betalväxeln, som i sin tur integrerar sitt system mot banker, kortföretag, operatörer och andra betaltjänsteaktörer. Varför en betalväxel anlitas beror på tidsbesparing och integrationskostnader, då det skulle ta för mycket tid att integrera de båda systemen så att de skulle vara funktionella. Men det skulle även besparas pengar då varje integration och uppdatering kostar pengar.

Det är dock inte helt gratis att använda en betalväxel, då det läggs på en transaktionsavgift ovanpå betaltjänsteaktörernas transaktionsavgifter. När en betalning sker med hjälp av en betalväxel lagras en konsuments kortnummer i krypterad form, men endast så länge betalningen inte är helt slutförd. I vissa fall visar en Internetbutik vilken betalväxel som används, då detta kan skapa en känsla av säkerhet för kunden som funderar på att handla varor.

Det skiljer en del mellan olika betalväxlar, vad för slags tjänster som erbjuds, vad tjänsterna kostar och vilken slags säkerhet som finns. För att få en överblick på vad en betalväxel erbjuder behandlas ett par exempel här nedan.

2.5.1.1 Dibs – DebiTech

Dibs och DebiTech är två betalväxlar som grundades i Danmark respektive Sverige men som gick ihop under sommaren 2006 för att skapa *”ökad konkurrensförmåga, starkare position på den skandinaviska marknaden och förbättrad position för fortsatt expansion och tillväxt.”* [41]

Dibs – Debitech tillhandahåller olika betaltjänster, och inte enbart för webbhandel. Men enbart webbhandelbetallösningar kommer att behandlas här. Då de två företagen gått samman nyligen finns ingen gemensam hemsida, vilket innebär att de fortfarande erbjuder olika betaltjänster till olika pris. De lösningar som finns är dock likartade och kommer därmed slås ihop för redovisande syfte. De alternativ för betallösningar som finns är:

- *Start*
  Detta är valet för säljaren som har behovet att komma igång med kortbetalningar över Internet så snabbt som möjligt. Det är skapat för företag som säljer produkter över Internet, och det är ett paket som är ett av de mest omfattande systemen men även ett av de mest prisvärda.

  Det ingår stöd för VISA och Mastercard, men om 3-D Secure ska användas måste det köpas till. Engångskostnaden för etableringen är 990 kr, det finns ingen månadsavgift men för varje transaktion tar betalväxeln 1%, lägst 290 kr.

[41] http://www.debitech.com/nyhetsarkiv/5.6433d6e310d82ee289d8000880.html (Acc. 2006-11-26)

- *Bas*
  Detta paket är för de som har lite större krav än vad Startpaketet kan erbjuda. Det har fler avancerade funktioner och kan integreras med i stort sett alla system. Det ingår stöd för de flesta betalningsformer, t.ex. Visa, Mastercard, Diners Club och American Express.

  Det går även att köpa till direktbetalning inom Sverige, samt fakturabetalningar. Med fler betalningsmöjligheter nås fler kunder och fler köp kan tänkas. Etableringskostnad är 3.990 kr, med en månadsavgift på 490 kr och transaktionsavgiften är 0.5% alternativt kan ett fast pris lämnas per transaktion.

- *Premium*
  Detta är valet för företag med stora behov kring betalningsmöjligheter. När detta paket ska integreras behövs personal från betalväxeln för att instruera hur integrationen ska ske. Därefter kontrolleras företaget varje kvartal så att det lever upp till skiftande lagstiftningsmässiga krav från kortinlösare.

  Paketet har stöd för alla vanliga kontokort, samt t.ex. Switch/Solo och Dankort/eDankort och det går att köpa till direktbetalning inom Tyskland, Norge, Finland, Danmark och Estland. Sverige ingår i paketet. Det ingår även en massa tillägg för administrationen och säkerheten. Både 3-D Secure och support ingår i paketet. Etableringskostnad är 9.990 kr, med en månadsavgift på 990 kr och transaktionsavgiften lämnas som offert.[42]

2.5.1.2 Samport

Samport är en svensk betallösning som tillhandahåller webbhandelbetallösningar samt betallösningar för fysiska butiker. På Samports hemsida står det att företaget erbjuder helhetslösningar såväl som kundanpassade system för betalning, administration och uppföljning, och allt med branschens högsta säkerhetskrav.[43]

Samport erbjuder två alternativ för webbhandel som de kallar för API-lösning och ”hostad” lösning. I kapitel 2.5.2.1 respektive 2.5.2.2 finns beskrivningar av dessa tekniker. Utöver dessa två alternativ erbjuds även tre tilläggspaket som lägger till tjänster såväl som ökad säkerhet.[44]

[42] http://www.dibs.dk/1644.0.html (Acc. 2006-11-26)
[43] http://www.samport.se/company.asp (Acc. 2006-12-01)
[44] http://www.samport.se/PDF/Samport%20Prislista%20-%20Sverige%20kortversion%20internet%2020060407.pdf (Acc. 2006-12-01)

- *E-handel - API-lösning*
  Insamling av transaktion, auktorisation, debitering samt leverans till bank eller inlösare ingår i denna tjänst. Samport har utvecklat verktyg som gör det lätt att hantera och följa upp transaktioner i realtid, dessa ingår givetvis. En egen betalserver krävs för en API-lösning.

  Anslutningsavgiften är 995 kr, månadsavgiften såväl som avgiften för 3-D Secure är helt gratis. Pris för varje transaktion är 0,50 kr.

- *E-handel - Hostad lösning*
  Precis som med API-lösningen ingår insamling av transaktion, auktorisation, debitering samt leverans till bank eller inlösare. Verktygen som hanterar och följer upp transaktioner ingår också. Den enda skillnaden gentemot API-lösningen är kostnaden för transaktioner. Orsaken till den höjda avgiften är att det är Samport som står för betalservern, och inte säljföretaget.

  Anslutningsavgiften är 995 kr, månadsavgiften såväl som avgiften för 3-D Secure är helt gratis. Den förhöjda avgiften för varje transaktion är 2 kr.

- *Tilläggspaket - Direktbetalning*
  I detta paket ingår direktkoppling mot banksystem i Norden, Tyskland, Holland och Österrike.

  Anslutningsavgiften är 995 kr, månadsavgiften är gratis. Avgiften för varje transaktion är baserad på transaktionsavgiften för det valda E-handelalternativet. D.v.s. transaktionsavgiften för direktbetalning är 2 kr om en hostad lösning är vald.

- *Tilläggspaket - Fakturatjänst*
  En tjänst vilket möjliggör sändning av fakturaunderlag till vald faktureringsleverantör.

  Anslutningsavgiften är 995 kr, månadsavgiften är gratis. Avgiften för varje transaktion är baserad på transaktionsavgiften för det valda E-handelalternativet. D.v.s. transaktionsavgiften för fakturatjänst är 2 kr om en hostad lösning är vald. Det tillkommer även en fakturaavgift om en pappersfaktura ska skickas ut, den är på 45 kr.

- *Tilläggspaket - Avancerat bedrägeriskydd SAFS GT/AI*
  Denna tjänst innebär att en bedrägerifunktion implementeras i den ursprungliga E-handelalternativet. Tjänsten varnar för uppenbara bedrägerier och det blockerar transaktioner gjorda från vissa länder, kort utgivna från vissa länder, t.ex. länder som inte ingår i 3-D Secure. Det ingår även en funktion som analyserar geografisk placering samt innehåll i ordern som kallas för SAFS GT/AI.

  SAFS kontrollerar bl.a. IP-nummer, Internetleverantör, kortutgivare samt land samtidigt som alla kortnummer matchas mot en aktuell svartlista.[45]

  Det finns ett rörligt pris för denna tjänst, såväl som ett fast pris. Det rörliga priset består av en anslutningsavgift på 1.995 kr med en månadsavgift som är gratis, och transaktionsavgiften kostar 1,50 kr. Det fasta priset för tjänsten består av en anslutningsavgift på 1.995 kr, där månadsavgiften får offereras. Pris per transaktion är dock helt gratis.

2.5.2 Kortbetalning

Kortbetalning över Internet brukar ske genom två olika varianter, API och ”hostad” lösning. Ett inlösenavtal krävs även om betalning ska kunna ske.

2.5.2.1 API-lösning

När en API-lösning används så betalar kunden till Internetbutiken i den virtuella kassan. Internetbutiken samlar in kortinformationen och strukturerar hela betalningen. En betalväxel validerar köpet i realtid för att sedan skicka tillbaks en svarskod huruvida köpet är godkänt eller ej. Efter det så sköts all hantering genom Internetbutikens system.

Detta ställer höga krav på systemets säkerhet och att all korthantering följer PCI-standarden. Denna standard står det mer om i kapitel 2.6.5. De stora fördelarna med att använda en API-lösning är att Internetbutiken har kontroll över hur betalningen sker och enkelt kan anpassa den.

[45] http://www.samport.se/service_safs.asp (Acc. 2006-12-01)

2.5.2.2 Hostad lösning

Vid en ”hostad” betalning så sköts allt ansvar för transaktionen utav betalväxeln. När betalningen sker så kontrolleras all information av betalväxeln och kortinformationen sker helt enligt PCI-standarden. Den stora fördelen med att använda en ”hostad” lösning är att företaget bakom Internetbutiken slipper stå för en server och SSL-certifikat.

2.5.2.3 Inlösenavtal

För att en betalning kan ske rent tekniskt måste butiken som ska sälja föremål över Internet skriva ett inlösenavtal. Med ett inlösenavtal kan butiken vara säker på att pengarna inte mellanlandar på ett annat konto först, eftersom avtalet talar om vart betalningarna ska gå. Ett inlösenavtal skrivs mellan butiken och en bank. Att förhandla med banken om priset är vanligt eftersom butiker kan vara väldigt olika, både i storlek och i omsättning. Ofta är det en startavgift för avtalet, sedan en månads- eller årsavgift och till sist en avgift per betalning från kund.[46]

Då kunden har valt sina varor och skrivit in sitt kontonummer, giltighetstid samt CVC2[47] så sker en kontroll att kortet är giltigt samt att köpet är godkänt. Betalväxeln som hanterar all information skickar alla transaktioner till banken via CEKAB, Centralen för Elektroniska Korttransaktioner AB.[48]

2.5.3 Direktbetalning

Denna betalningsmetod kan endast användas av kunder som är anslutna till en Internetbank. Ett försäljningsavtal måste tecknas med bankerna som ska anslutas till direktbetalning.

Då kunden anger att köpet ska genomföras med direktbetalning så skickas kunden vidare till sin Internetbank. Samtidigt som detta sker, så sammanställs ordern och skickas till betalväxeln för att behandlas och sedan skickas vidare till kundens bank. Efter det skriver kunden in sina uppgifter i Internetbanken och godkänner köpet. Efter att köpet är godkänt så skickar banken uppgift om köpet till betalväxeln. Och till sist skickar betalväxeln status om köpet till Internetbutiken.

[46] http://www.handelsbanken.se (Acc. 2006-12-03)
[47] CVC2 och CVV2 är ett kryptografiskt framtaget värde som kan härledas till kortet och är avsett att förhindra förfalskning. Det är Mastercard respektive Visa som har denna benämning på systemet.
[48] CEKAB etablerades som bolag 1989 och arbetar på uppdrag av banker och finansinstitutioner. Ägare är Handelsbanken, Swedbank, Nordea och Danske Bank. År 2005 hanterades cirka 959 miljoner transaktioner. Officiell hemsida: http://www.cekab.se

En stor fördel med att erbjuda direktbetalning är att de kunder som är rädda för att ge ut kortnummer över Internet inte utesluts. Att slutföra köpet med sin bank kan kännas tryggare, både att ha den som mellanhand men även att kunden är van vid sin banks gränssnitt.

2.5.4 Kontoöverföringar

Betalningar över Internet har på senare blivit mer och mer vanliga. De har inte samma lagar som om t.ex. banker skulle göra samma betalningar. De är inte lika skyddade då de inte har samma lagar, och har därmed en gräns på några tiotusentals kronor. Idag kan endast transaktioner ske på konton inom en och samma betaltjänst. Men överföringar mellan betaltjänster kan förverkligas, antingen på liknande sätt som traditionella banker eller genom en tredje part.

Kontoöverföringar är ganska okomplicerade, användare kontaktar betaltjänsteleverantören genom SSL och skriver in belopp samt vilket konto som pengar ska föras över till. En extra funktion som finns hos vissa leverantörer är att användare kan skriva in att överföringen ska ske en viss tidpunkt.

2.5.5 Engångskortnummer

Om en kund inte vill ge ut sitt kortnummer på Internet så kan vissa kort kopplas till engångskortnummer. Denna tjänst tillhandahålls av kundens bank, och den lanserades år 2000. Kunden laddar ner ett litet program till sin dator som skapar en säker kontakt med bankens server. Med hjälp av programmet kan kunden skapa ett kort med valfri summa, hur många gånger kortet kan användas samt en giltighetstid. Så när kunden har funnit en vara och ska betala den med kort, så anges kortnumret som skapades med programmet. Om kunden angett att kortet endast kan användas en gång, så spärras kortet efter köpet.

Det har dock skapat en del problem, då en del affärer tar en liten summa pengar först för att se så att kortet faktiskt finns, för att sedan, en andra gång, ta hela summan. Banken tror att köpet genomförts när affären tar pengar första gången, för att sedan spärra kortet. Då kan affären inte ta hela summan, och köpet kan ej genomföras. Så lösningen kan vara den att förlänga giltighetstiden samt välja att kortet kan användas för multipla köp.

En av fördelarna med denna funktion är att kundens riktiga kortnummer inte ges ut på Internet. En annan är att affären tror att engångskortet är ett helt vanligt kort, och därmed inte behandlar betalningen annorlunda på något vis. De största nackdelarna skulle nog vara att kunden måste gå till sin bank för att anskaffa denna tjänst, samt att det tar tid att starta upp programmet, för att sedan fylla i alla uppgifter som behövs för att få ett engångskortnummer.

2.5.6 Digitala plånböcker

Digitala plånböcker kallas tjänster på Internet där kunden skapar ett konto och för över pengar till det för att kunna handla över Internet. På det kontot loggar kunden sedan in med användarnamn och lösenord på liknande sätt som på hemsidor för e-post. Kunden laddar upp den summa den själv vill till kontot. På det viset kan kunden reglera hur mycket pengar som ska användas på tjänsten. De webbutiker som har ett avtal med en digital plånbok får en länk på deras sida till betalningstjänsten. Kunden klickar på länken för att betala via plånboken och skickas då till inloggningen för den. Därifrån förs tillsist så mycket pengar som behövs över. Det finns flera olika varianter av digitala plånböcker.

2.5.6.1 Paypal

Paypal är ett Ebay-företag som grundades 1998.[49] Den 13 februari 2006 överskred PayPal 100 miljoner användare i 55 länder och regioner världen över. Tanken med PayPal är att göra finansiella transaktioner mellan datorer på ett säkert sätt med hjälp av mjukvara som krypterar information. Det används främst till att betala auktioner online, vid köp av varor och tjänster över Internet och vid donationer.

Ett vanligt PayPal-konto är gratis att skaffa och kunden behöver inte ange sitt bankkonto, även om ett checkkonto eller kreditkort är nödvändigt för flera av PayPals funktioner. Vid registrering av ett konto finns det tre olika varianter att välja mellan.

- *Personal Account*
  Ett konto som passar den som bara ska handla varor på Internet via auktion eller Internetbutik. Det accepterar dock inte kreditkortsbetalning.

- *Premier Account*
  För den som både vill köpa och sälja varor över exempelvis Ebay eller annan Internetauktion. Det godkänner alla typer av betalningar.

- *Business Account*
  Ett konto för företag som ska sälja varor eller tjänster, samma som Premier, men det går att ha flera användare.

[49] http://www.paypal.com (Acc. 2006-09-22)

PayPal fungerar som en mellanhand mellan bankkonton och kreditkort och hjälper till att få överföringen att ske på ett säkert sätt. När en handlare accepterar en betalning från ett kort, betalar han först en liten avgift på ungefär tio cent plus omkring två %. Den avgiften är uppdelad av några mindre avgifter som betalas ut till alla olika företag som har del av transaktionen. Handlarens bank och företagen som är inblandade i kreditkortet t.ex.

Eftersom all information, såsom kreditkorts- och bankkontouppgifter och adresser, stannar hos PayPal, slipper köparen och säljaren ge varandra uppgifter. På så sätt fungerar PayPal som en säkerhet. Alla pengar som finns i konton på PayPal ligger i ett eller flera bankkonton där PayPal tar ut ränta på dem.

2.5.6.2 Paynova

Paynova är en digital plånbok som det finns stöd för på hundratals nätbutiker, i Sverige såväl som utomlands. För att fylla på plånboken kan användaren använda plus- och bankgiro, kreditkort eller överföra pengar via en Internetbank.

Att skaffa Paynovaplånboken är helt gratis, likaså att betala med den och att fylla på med belopp på 99 kr eller mer. Fyller kunden på med mindre än 99 kr drar de en så kallad småbeloppsavgift på fem kr. Om kunden sedan vill ta ut pengar igen tar de en uttagsavgift på fyra % av beloppet, dock minst 20 kr.

Paynova håller all information om användaren krypterad i ett säkert system som är skyddat mot intrång, och försäljningsstället kan inte ta del av användarens uppgifter genom Paynovaplånboken.[50]

2.5.6.3 Moneybookers

Moneybookers tillhandahåller företag och konsumenter med en e-postadress, en säker och kostnadseffektiv tjänst att skicka och ta emot betalningar över Internet. Användaren slipper ge ut sin kreditkortsinformation varenda gång ett köp genomförs, och allt som behövs är mottagarens e-postadress för att skicka pengar. Om en person fått pengar skickade till sin e-postadress, räcker det för denne att öppna ett konto hos Moneybookers med den adressen, sedan har den pengarna på det kontot.

Moneybookers fungerar i alla länder, och har dessutom lokala betalningsmetoder i 30 länder, däribland alla nordiska länder såsom svenska bankgiro. Överföringen av pengar sker direkt i realtid, även mellan länder.

[50] http://www.paynova.com/swe/personal/wallet/functionality.asp?icp=8D42980C67130C3A (Acc. 2006-09-22)

Moneybookers har funnits som företag sedan 2002 och hade 2006, över två miljoner användare världen över. Deras vision är att vara världens ledande destination för betalningar online. De tycker själva att de är väl lämpade för små företag, säljare online och alla andra som inte tycker att det räcker att sälja i traditionell, fysisk butik

Det är gratis att registrera sig, ladda upp pengar från bank, få pengar och be om pengar. Däremot kostar det att ladda upp pengar från ett kreditkort, 2,5% av summan; att skicka pengar, 1% och högst 0,50 Euro; att föra över pengar till bank, 1,80 Euro; och att föra över pengar via check, 3,50 Euro.[51]

2.5.6.4 Neteller

Neteller har erbjudit säkra transaktioner online sedan 1999 och har nu över 2,5 miljoner användare. Neteller är en digital plånbok som låter deras användare ladda, ta ut och skicka pengar. Kunder kan skicka pengar till och från vilket företags hemsida som helst som har stöd för Neteller, och till och från andra Netellerkunder.

Enda gången användaren behöver ange sina kreditkorts- eller bankuppgifter är när användaren skapar kontot hos Neteller, sedan görs säkra transaktioner över Internet på de hemsidor som stödjer Neteller. Företagen användaren handlar hos har ingen tillgång till kontouppgifterna, de kommer bara åt Neteller, för att hindra obehöriga transaktioner att äga rum, samt att skydda personlig information.[52]

2.5.6.5 Payson

Payson sköter betalningar inom sitt system, d.v.s. mellan två Paysonkonton. Användarens konto har olika funktioner som underlättar för funktionaliteten:

- *Fyll på ditt konto*
  Visa, Mastercard eller Internetbank kan användas. Ange vilken summa som ska sättas in och varifrån de ska dras.

- *Uttag*
  Användaren kan ta ut pengar från sitt Paysonkonto till sitt bankkonto.

- *Överföring*
  Fyll i vilken summa som ska betalas och mottagarens e-postadress. Efter det görs en betalning till ett annat Paysonkonto.

[51] https://www.moneybookers.com (Acc. 2006-09-22)
[52] http://www.neteller.com/ (Acc. 2006-09-22)

- *Begära pengar*
  Via ett mailformulär kan en säljare skicka en elektronisk faktura till en kund. Kunden kan sedan enkelt betala genom att klicka på en länk i ”begära pengar”-mailet.

Payson fungerar i större delen av Europa (29 länder). Payson har framförallt två fördelar:

- Payson är optimerat för Internet genom att användaren kan skicka pengar till mottagarens e-postadress, inget kontonummer behövs samt att kunden både kan betala och få betalt med Visa/Mastercard, Internetbank eller Bankgiro, men vilket idag endast erbjuds till företag.

- Paysongaranti gör att en användare kan känna sig säker att betala och få betalt på Internet. Säljaren får pengarna innan varan skickas och köparen kan kontrollera varan innan Payson släpper pengarna till säljaren. Det är en helt unik lösning som endast Payson erbjuder.

Som användare registreras e-postadressen, därefter kan pengar både tas emot och skickas. Payson mailar användaren direkt när någon betalat till kontot. En användare kan skicka pengar till någons e-postadress även om mottagaren inte är medlem hos Payson. Mottagaren kan efter att pengarna skickats registrera sig på Payson och få tillgång till pengarna.[53]

2.6 Säkerhet

2.6.1 SSL/TLS

När en kund ska betala med ett kreditkort eller kontokort så används troligtvis Secure Socket Layer (SSL). Kortinformation som skickas över Internet kan ses som en icke säker transmission. Men genom att använda SSL kan informationen skyddas genom kryptering men det kan även verifiera identiteten hos både kunden och säljaren. Det som sker oftast idag är att säljaren använder SSL för att verifiera sin identitet. SSL tillämpas i de flesta webbservrar och Internetläsare.

När SSL används, så är det ofta i kombination med webbprotokollet http. Vid t.ex. inloggning till en Internetbutik så börjar adressen med https:// vilket betyder att det är en SSL-krypterad session. SSL kan användas för andra protokoll, t.ex. postprotokollen SMTP och POP. Idag används oftast en 128-bitars kryptering. Innan år 2000 användes SSL med 40-bitars kryptering, vilket inte tar lång tid att knäcka och var därmed ett dåligt skydd. Det går att köpa ett 256-bitars certifikat om det behövs, t.ex. vid många kunder och stora order.

[53] https://www.payson.se/prod/default.aspx (Acc. 2006-09-22)

Högnivåkryptering, såsom 128 bitar, kan beräkna 2^{88} gånger så många kombinationer som 40-bitars kryptering. En hacker som skulle kunna knäcka en 40-bitars kryptering på några få minuter skulle med samma verktyg behöva en triljon år för att tränga in i en session som skyddas av ett certifikat med en Server-Gated Cryptography (SGC) funktion.[54]

SSL har utvecklats till en ny Internet-standard och kallas Transport Layer Security (TLS) av IETF[55]. TLS är baserad på SSL version 3.0. I en del applikationer är det möjligt att välja om SSL eller TSL ska användas, då de är inkompatibla med varandra. Men det är möjligt att ändra om TLS till SSL version 3.0 om det skulle behövas. Det har inte skett så mycket utveckling inom TLS-standarden, vilket har gjort att SSL fortfarande används väldigt ofta.

Lite kort om hur SSL fungerar

Klienten skickar och tar emot ett antal olika strukturer av handskakningar.

- Först skickas ett *ClientHello* meddelande som specifierar certifikat, komprimeringsmetoder och högsta protokollversion som stödjs. Några slumpmässiga bytes skickas samtidigt, vilka kommer användas senare.
- Sedan tar klienten emot ett *ServerHello*, där servern väljer anslutningsparametrarna av valen som skickades innan av klienten.
- När parametrarna är kända så byter klienten och servern certifikat med varandra. Dessa certifikat är för nuvarande X.509[56], men även OpenPGP[57] baserade certifikat används.
- Servern kan begära ett certifikat från klienten, så att anslutningen blir gemensamt säker.

Klienten och servern bestämmer en gemensam hemlighet som kallas “master secret”, detta skulle kunna vara resultatet från ett Diffie-Hellman[58] utbyte. All annan nyckeldata tas från denna “master secret” som sedan går igenom en noggrant konstruerad "Pseudo Slumpmässig Funktion".

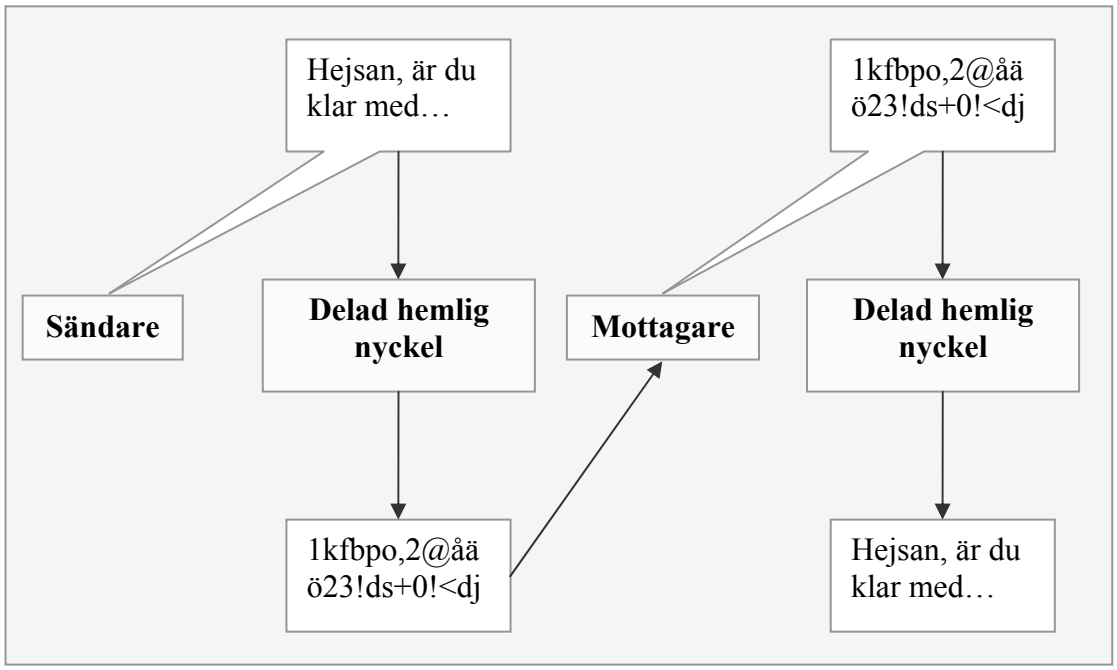
[54] http://www.verisign.se/products-services/security-services/ssl/ssl-information-center/strongest-ssl-encryption/index.html (Acc. 2006-09-22)
[55] IETF (Internet Engineering Task Force) är en organisation som anordnar diskussioner och överenskommelser om Internetteknik. Officiell hemsida: http://www.ietf.org
[56] En standard som beskriver certifikatet som används för t.ex. säker Internetöverföring. X.509-certifikat är en del av en PKI och verifieras av en CA hörande till en vald PKI.
[57] OpenPGP är en s.k. öppen standard av en äldre version som enbart hette PGP (Pretty Good Privacy). PGP anses vara en världens mest använda system, men på grund av patentproblem skapades den nya OpenPGP. Officiell hemsida: http://www.openpgp.org
[58] Diffie-Hellman möjliggör så att två klienter som inte har någon tidigare kännedom om varandra, kan tillsammans åstadkomma en delad hemlig nyckel över en osäker kommunikationskanal.

2.6.2 Kryptering och dekryptering

Kryptering är ett begrepp som går ut på att ett meddelande i klartext omvandlas till kryptotext, vilket är oläsbart. Krypteringen sker genom ett kryptosystem (encryption system eller cryptographic system) samt en tillhörande nyckel. Dekryptering är motsatsen, processen omvandlar kryptotext till klartext med en specifik nyckel. Det finns i huvudsak två olika metoder att använda.

2.6.2.1 Symmetrisk kryptering

Vid denna process används samma krypteringsalgoritm och en identisk nyckel för både kryptering och dekryptering. För att metoden ens ska fungera så måste både sändare och mottagare dela på den hemliga nyckeln. Sändaren krypterar sitt meddelande med nyckeln för att sedan skicka iväg meddelandet. Mottagaren måste använda samma nyckel för att en dekryptering ska kunna ske. Det är vanligtvis denna metod som SSL använder.[59] Figur 6 visar hur symmetrisk kryptering fungerar.



Figur 6 Hur symmetrisk kryptering fungerar[60]

[59] http://www.windowsecurity.com/articles/SSL-Acceleration-Offloading-Security-Implications.html (Acc. 2006-12-07)
[60] Handbok i IT-Säkerhet, s. 100, Predrag Mitrovic, Pagina Förlags AB 2001, Göteborg ISBN 9163606747

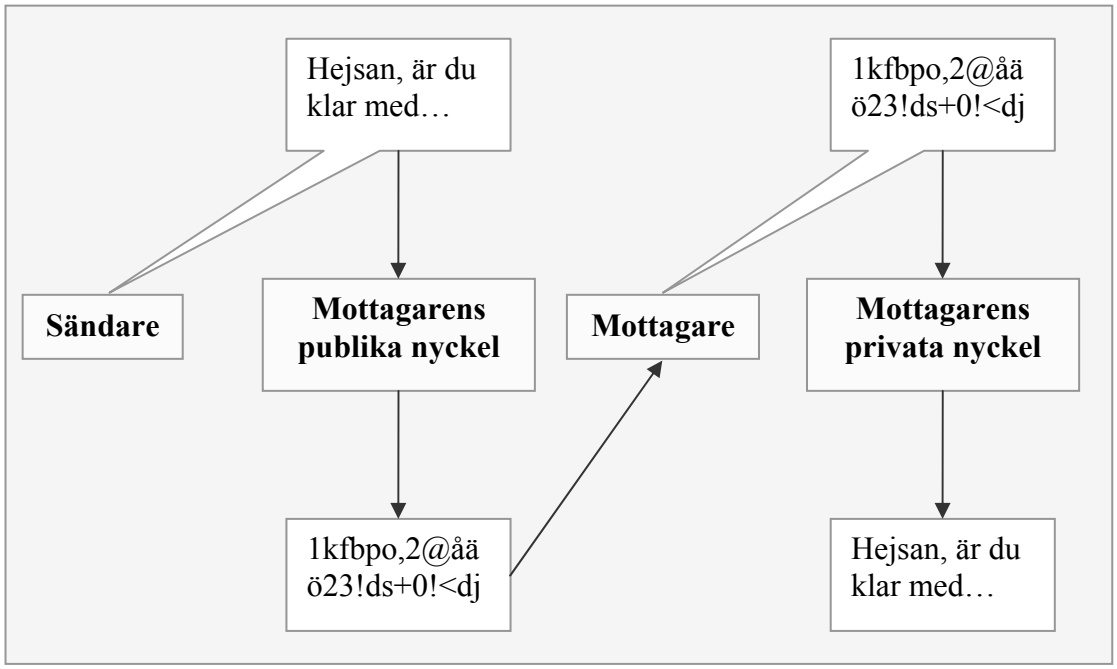
Symmetrisk kryptering används ofta inom militären och intelligenstjänster då denna typ av kryptering anses mer svårforcerad än den asymmetriska krypteringen. Då symmetrisk kryptering är mindre processorkrävande än asymmetrisk så kan större informationsmängder hanteras med samma datorkapacitet, och detta är en fördel när det handlar om transaktioner som ska ske snabbt över Internet.

En av de största nackdelarna med symmetrisk kryptering är att det är väldigt svårt att sköta en nyckeladministration på ett globalt och distribuerat system. Dessutom bör nyckeln bytas ut med jämna mellanrum för att hålla den hemlig, för alla som har tillgång till nyckeln kan läsa allt i klartext. Ju mer data som någon har som är krypterad med samma nyckel, desto lättare är det att bryta krypteringen.

2.6.2.2 Asymmetrisk kryptering

Vid asymmetrisk kryptering används samma krypteringsalgoritm men olika nycklar används för kryptering och dekryptering. Det används även ett nyckelpar som kallas för privat- och publik nyckel. Den privata nyckeln är hemlig för alla, medan den publika nyckeln är öppen och kan läsas av alla.

Sändaren krypterar meddelandet med mottagarens publika nyckel, som därefter skickar iväg meddelandet. Mottagaren tar emot meddelandet och dekrypterar det med sin privata nyckel. Figur 7 visar detta förlopp.



Figur 7 Hur asymmetrisk kryptering fungerar[61]

[61] Handbok i IT-Säkerhet, s. 101, Predrag Mitrovic, Pagina Förlags AB 2001, Göteborg ISBN 9163606747

Det finns alltid ett nyckelpar som hör ihop, det är detta som möjliggör denna kryptering och dekryptering. Utifrån den privata nyckeln är det möjligt att räkna ut den publika, men det är omöjligt att räkna ut den privata genom den publika. Informationen kan krypteras och dekrypteras i vilken ordning som helst, men en enda nyckel kan inte användas för kryptering och dekryptering.

Privata nycklar lagras vanligtvis på en dators hårddisk eller i ett smart-card, medan publika nycklar tillhandahålls via en elektronisk katalogtjänst (PKI). I kapitel 2.6.6 beskrivs dessa katalogtjänster mer ingående.

2.6.3 SET

SET (Secure Electronic Transaction) skapades av bl.a. Mastercard, Visa, IBM, Netscape och Microsoft för att skydda båda parter i ett genomfört köp. Trots att systemet har ett brett stöd så har det inte blivit spritt, då det är komplicerat att använda. För att hantera elektroniska certifikat behövs en fungerande infrastruktur, då certifikatet stödjer kryptering, autentisering och revokering. SET kan använda 128-bitars kryptering.

Vad detta betyder är att kunden inte kan förneka en betalning, kundens kortinformation inte visas för betalningsmottagaren och att kunden ser att betalningsmottagaren är den som den utger sig för att vara. Då SET är så komplicerat och kräver omfattande tekniska lösningar såsom certifikat hos kunden, så används inte detta idag, och det kommer troligtvis aldrig att användas. Det som används istället för SET kallas 3-D Secure.

2.6.4 3-D Secure

Mastercard och Visa skapade 3-D Secure som ska ge både kund, betalningsmottagare samt kortföretag såsom inlösande banker trygghet. Tryggheten består av att betalningar ej kan bestridas genom t.ex. bedrägeri samt att endast kortinnehavaren (kunden) kan använda kortet för betalningar. Precis som SET kräver 3-D Secure en del certifikat, dock ej lika många. Det som krävs är att den säljande parten har en viss programvara.

3-D Secure ska lösa i stort sett alla problem precis som SET ska göra, såsom rutiner, policies och befintliga säkerhetslösningar som t.ex. SSL/TSL. När kunden ska betala för en vara så verifieras först och främst kortinnehavarens identitet, eller en koll ifall en annan person får använda kortet genom en gemensam hemlighet, såsom ett lösenord alternativt PIN-kod. Efter denna verifikation så informerar kortutfärdaren ifall betalningsmottagaren kan acceptera betalningen.[62]

[62] http://partnernetwork.visa.com/pf/3dsec/main.jsp (Acc. 2006-09-22)

2.6.5 PCI

PCI (Payment Card Industry Data Security Standard), även känt som PCI-DSS, är en säkerhetsstandard som har enats av bl.a. Visa och Mastercard. Den beskriver hur alla kortnummer och annan transaktionsinformation ska hanteras.

Den säger att ett företag ska:

- undvika lagring av kortinformation eller annan känslig information
- radera information som inte längre används
- behörighetsskydda tillgången till kortinformation med användaridentiteter och lösenord
- säkerställa att utgivna behörigheter inte sprids till obehöriga
- säkerställa att användandet av behörigheter kan spåras
- säkerställa interna rutiner för att undvika insiderbrott eller externa intrång i systemet
- installera och underhålla säkerhetsprogramvara och skydda systemet mot datavirus
- genomföra regelbundna tester av säkerhetssystem och dess hantering
- säkerställa att inte hela kortinformationen i kortets magnetspår och kortets säkerhetskod (de tre sista siffrorna tryckta i signaturfältet på kortets baksida) lagras efter avslutad kortbetalning
- säkerställa att kortnummer alltid trunkeras, det vill säga aldrig trycks i sin helhet på kvitton eller annan tryckt media.[63]

2.6.6 Public Key Infrastructure och Certificate Authority

PKI (Public Key Infrastructure) står för den infrastruktur som hjälper användningen av elektroniska signaturer. Det finns olika funktioner i en PKI, såsom

[63] http://www.foreningssparbanken.se/sst/www/inf/out/fil/0,,261568,00.pdf (Acc. 2006-11-17)

- *Certifiering*
  Certifieringen utförs av en Certificate Authority (CA), som t.ex. Geotrust[64]. För att anskaffa ett certifikat måste en del punkter gås igenom. Först och främst ska användaren skapa en certifieringsansökan, Certificate Request, som innehåller administrativa uppgifter och användarens öppna nyckel. Efter det så godkänns ansökan och ett intyg utfärdas, kallat certifikat.

  Certifikatet är det digitala dokumentet som är signerat med CA:ns egna nyckel. Det är även ett bevis på en koppling mellan ett nyckelpar (består av en publik nyckel som alla kan ta del av, samt en privat nyckel som är hemlig) och en person, ett företag eller ett datorsystem. Certifikat är tidsbegränsade och upphör att gälla efter en bestämd tid. Då ska användaren byta ut sina nycklar och få tag på ett nytt par genom certifiering.

- *Verifiering*
  När en verifikation sker, används en certifieringsrutin som på engelska kallas Certificate Policy Statement (CPS). I den beskrivs vilka rutiner som ska gås igenom samt hur nycklar hanteras. Det är ett viktigt dokument när det gäller att avgöra hur bra tjänster en CA tillhandahåller.

  Det finns olika typer av certifiering och en CA kan ha många olika, alla med olika värde. En viss certifiering kan verifiera koppling mellan en fysisk person och en nyckel, medan en annan kan verifiera koppling mellan en e-postadress och en nyckel.

  SSL använder denna tjänst för att skapa säker kommunikation i kombination med protokollet http. Certifikatet verifierar att en viss nyckel en server använder tillhör just den servern samt det speciella företaget som äger servern. Nyckeln som blir verifierad används sedan utav SSL.

- *Revokering*
  Vid speciella fall kan en CA återkalla en certifiering, om detta händer bör informationen spridas snabbt så att användningen utav nycklarna slutar. Detta skulle t.ex. kunna ske om en anställd på ett företag har slutat och inte längre får använda verktygen som fanns på jobbet.

Då det är fråga om en elektronisk handling, vars innehåll är ett flertal olika uppgifter, så är certifieringen väl specificerad och standardiserad. Standarden som används är oftast X.509. I kapitel 2.6.1 beskrivs denna teknik kortfattat.

[64] Geotrust är världens näst största företag inom digitala certifikat. Officiell hemsida: http://www.geotrust.com

Uppgifterna består utav namn, adress samt annan nödvändig administrativ data. På detta läggs den publika nyckel som tillhör användaren. När det gäller uppgifterna från en CA så läggs även dessa in. De brukar vara administrativa data, hur kontroll av certifieringen kan ske, hur länge certifieringen är giltig samt en länk till deras hemsida.

Resultatet, efter att CA:n signerar med sitt eget nyckelpar, är ett certifikat som kan spridas fritt för att koppla en nyckel till t.ex. en person eller företaget. Certifikatet kan kallas för en bas, den är själva kopplingen mellan ett nyckelpar som kan användas för identifiering, kryptering, säker e-post eller som ett juridiskt objekt. Värdet för ett certifikat är självklart väldigt viktigt, då mottagaren av ett certifikat måste ha tillräckligt med förtroende för den utfärdande CA:ns tjänster innan användning utav den publika nyckeln i certifikatet kan godkännas för kommunikation.

CA kan även delas in i två grupper, interna och externa. Ett exempel på interna CA:n kan vara en bank som har en egen CA-tjänst. Banken använder den tjänsten för signering av nycklar för kunder som använder Internet-tjänster. Eller t.ex. ett företag kan använda tjänsten för anställda som kan tänkas använda nyklarna i arbetet. Men även statliga myndigheter såsom Tullverket kan vara en CA, de använder tjänsten bl.a. för hanteringen av elektroniska tullhandlingar.

Om ett företag inte har en egen CA, så kan ett företag som har certifieringstjänster anlitas. Det är detta som kallas för en extern CA. Det används framför allt för webbservercertifikat (SSL-krypteringen), men även för e-post (S/MIME-certifikat). De största och mest kända CA-företagen är Geotrust och Verisign[65]. Verisign kommer även att köpa upp Geotrust vilket kommer att skapa världens största företag inom digitala certifikat.[66]

2.6.7 Identitetsstöld och lösenord

Identitetsstöld förekommer ibland när någon tar ett kreditkort, bankkonto eller personnummer och handlar varor eller lånar i en annan persons namn.

Detta är ett växande problem i och med att köp över Internet och kreditkort används mer och mer i vårt samhälle. Därför är det också väldigt viktigt att man skyddar sin personliga information väl.

[65] Verisign är precis som Geotrust ett av de största företagen inom digitala certifikat till webbhandel. Officiell hemsida: http://www.verisign.com

[66] http://news.netcraft.com/archives/2006/05/17/verisign_to_buy_geotrust_combining_top_ssl_providers.html (Acc. 2006-11-26)

Industrin har utvecklat teknologier som förvränger viktig information, som t.ex. ett kreditkortsnummer, så att bara försäljaren användaren köper av och kreditkortsutfärdaren kan läsa den. Detta försäkrar att betalningsinformationen inte kan bli läst av någon annan eller ändrad på vägen.

Det finns flera sätt att säkerställa att en användare har det skyddet när betalningar genomförs över Internet. Ett av dessa sätt är att se om adressen till sidan som frågar om kreditkortsinformationen är säker, dvs. börjar på ”https”.

Många företag erbjuder även beställningar över Internet där kunden ger ut sina kontouppgifter över telefon. Om man som kund gör det, ska man anteckna telefonnummer, företag, datum och tid för samtalet, och namnet på personen som mottog informationen.

Men även detta behöver inte vara nog för att vara säker på att företaget är äkta, det enda som är helt gratis som kan skydda köpare på Internet, är omdömet. Om en vara kostar en struntsumma av originalpriset, undvik att köpa den. Ganska ofta har det hänt att personer har betalat för en produkt, för att sedan varken se pengar eller produkt.

När kunden sedan ska välja lösenord eller PIN-kod till diverse saker så finns det vissa intressanta knep:

- **Olika lösenord**
  Lösenord är väldigt viktiga vid användandet av datorer för att se till att ingen använder en användares dator eller personliga information utan tillstånd. Ett tips kan vara att använda sig av olika lösenord till olika aktiviteter, t.ex. ett lösenord till att logga in på nätverket, och ett för att skicka en order på Internet. Att ha ett speciellt lösenord till särskilt känslig information kan också vara att rekommendera.

- **Hur man väljer lösenord**
  Kunden bör helst inte använda sin adress, personnummer, telefonnummer, igenkännbara ord eller lätta sifferkombinationer som t.ex. 12345. För effektivast lösenord, använd en sträng av minst fem tecken. Blanda även versaler och gemener. För att lättare minnas ett lösenord kan man ta första bokstaven från varje ord i ett uttryck eller sångtext, och lägga till någon siffra och annat tecken. T.ex. ”tMottObg!5” får man fram från frasen ”Take Me Out To The Old Ball Game.”[67]

- **Hur man sparar lösenord**
  Skriv inte ner något lösenord i närheten av en dator där någon kan se det. Bär det heller inte i plånboken. Om en anteckning ändå krävs, ändra då ordningen på tecknen eller byt ut något tecken. På så sätt har den som skulle hitta lappen ändå inte hittat det exakta lösenordet.

[67] http://www.safeshopping.org/home.shtml (Acc. 2006-09-22)

- **Vem vill ta reda på någons lösenord?**
  Var väldigt försiktig med att svara på e-post, telefonsamtal eller brev som ber om lösenord, personnummer, bankkontoinformation, kreditkortsnummer eller annan personlig information. Säljare och finansiella institutioner frågar vanligtvis inte om sådan information.

Identitetstjuvar kan skriva e-post som ser precis ut som att de är från riktiga hemsidor. Därför ska användare vara på sin vakt och gärna kolla upp innan de svarar på ett sådant e-post. För att kontrollera att den person som kontaktar någon verkligen jobbar för företaget kan man som kund ringa och fråga efter personen. Lösenord ska inte behöva användas mer än när inloggning sker på användarkontot, ditt kreditkortsnummer ska inte ges ut mer än när en order verkligen läggs, och om uppgifter måste ges ut, gör det inte i ett vanligt e-post, utan via en säker webbplats.

2.6.8 Privatliv på Internet

Onlineförsäljare får ta in och spara namn, adresser och information om vilka sidor någon besöker, vilka produkter som köps, när de köps och vart de fraktas. Sedan kan säljaren dela med sig av den informationen med andra företag, eller sälja den. Som följd av detta kan kunder på sidan få mer spam eller telefonsamtal från telefonförsäljare.

Det privata är en vital del i att våga göra affärer online. Ju säkrare kunden känner sig med att dela med sig av sin information, desto mer vågar han göra affärer. Därför har Internetsäljare blivit uppmuntrade att skriva ner en ”privacy policy” på hemsidan. En säljares policy ska visa vilken information säljaren sparar om kunder, hur säljaren kommer använda informationen om kunden och huruvida kunden kan hoppa av om villkoren är dåliga.

När affärer ska ske med en Internetbutik, kan det vara bra att läsa genom dess policy först, om företaget inte har någon alls bör man som kund fundera en extra gång om man verkligen vill genomföra ett köp och lämna ut uppgifter. Om företaget har en policy finns det säkerligen en länk till den på deras hemsida, antingen som egen länk ”privacy policy” eller som en del under ”terms and conditions” eller ”legal terms”.

3 Genomförande

Efter inledande teoretiska studier påbörjades arbetet med att utveckla e-handelslösningen vilket det här kapitlet behandlar. Fokus ligger främst på implementation av en betallösning och säkerheten kring denna, men även delar av webbplatsens konstruktion behandlas.

För att få förståelse för hur användaren upplever säkerhet med avseende på kortbetalningar på Internet har även en enkät tagits fram. Hänsyn till resultatet av denna enkät har tagits vid implementationen av betallösningen.

Första steget i genomförandet var att ta fram en kravspecifikation, som har legat som grund i utvecklandet, tillsammans med uppdragsgivaren. Utifrån den färdigställda kravspecifikationen strukturerades sedan databasen upp och programmeringen av applikationen påbörjades.

3.1 Kravspecifikation

3.1.1 Allmänt

Eftersom e-handelsplatsen ska utvecklas till ett företag som till stor del handlar med kläder så är det viktigt att applikationen tillsammans med databasen anpassas för detta. Applikationen ska således kunna hantera produkter med olika attribut som till exempel storlek och färg. Dessa produkter med olika attribut måste behandlas med individuell lagerhantering.

Företaget har för avsikt att i framtiden expandera till marknader utanför Sverige. Därför krävs det av applikationen att kunna hantera olika språk och fraktalternativ till respektive land eller region. I startskedet är dock endast Sverige aktuellt men applikationen ska vara förberedd för expandering.

Butikens målgrupp anses vara framgångsrika män i åldrarna 20-45 år med relativt hög datorvana. Applikationen ska anpassas efter målgruppen med avseende på tillänglighet och funktionalitet.

3.1.2 Tekniska krav

3.1.2.1 Databas

All kund- och produktrelaterad information ska sparas i en Microsoft SQL Server 2005 databas. Lagrade procedurer ska uteslutande användas för att hämta och spara data i databasen.

3.1.2.2 Webbserver

Webbplatsen ska ligga på en webbserver med IIS 5.0 med Microsoft .Net Framework 2.0 installerat. Det ska finnas möjlighet att installera SSL-certifikat.

3.1.3 Säkerhet

Webbplatsen ska ligga på ett webbhotell med höga krav på säkerhet. Detta är viktigt då butiken kommer att spara personuppgifter i databasen. Daglig backup är även det viktigt för att inte förlora information om beställningar och dylikt.

Kortbetalningar ska ske krypterat via SSL för att skydda känsliga kortuppgifter. Finns stöd för 3D Secure eller andra bedrägeriskydd så ska dessa tillämpas. Även känslig information som lösenord ska krypteras i databasen för att förhindra att obehöriga kan komma över detta. Användaren ska informeras väl angående hur personuppgifter lagras och om säkerhet kring kortbetalningar.

För att följa PCI (Payment Card Industry Data Security Standard) standarden ska inga transaktionsuppgifter eller kortuppgifter lagras i databasen utan dessa uppgifter ska uteslutande hanteras av betalväxeln (Samport).

3.1.4 Icke funktionella krav

3.1.4.1 Upplösning

Målgruppen anses ha tillgång till skärmar med en upplösning av minst 1024 x 768. Därför får den maximala bredden på butikens hemsida ej överstiga 950 bildpunkter. Av layoutmässiga och estetiska skäl ska bredden vara statisk och sidan ska centreras i webbläsarfönstret.

3.1.4.2 Webbläsare

Webbplatsen ska fungera tillsammans med de moderna webbläsare som följer W3C-standarden inklusive Internet Explorer. Sidan ska visas utan layoutmässiga och användbarhetsrelaterade problem i både PC- och Mac-miljö.

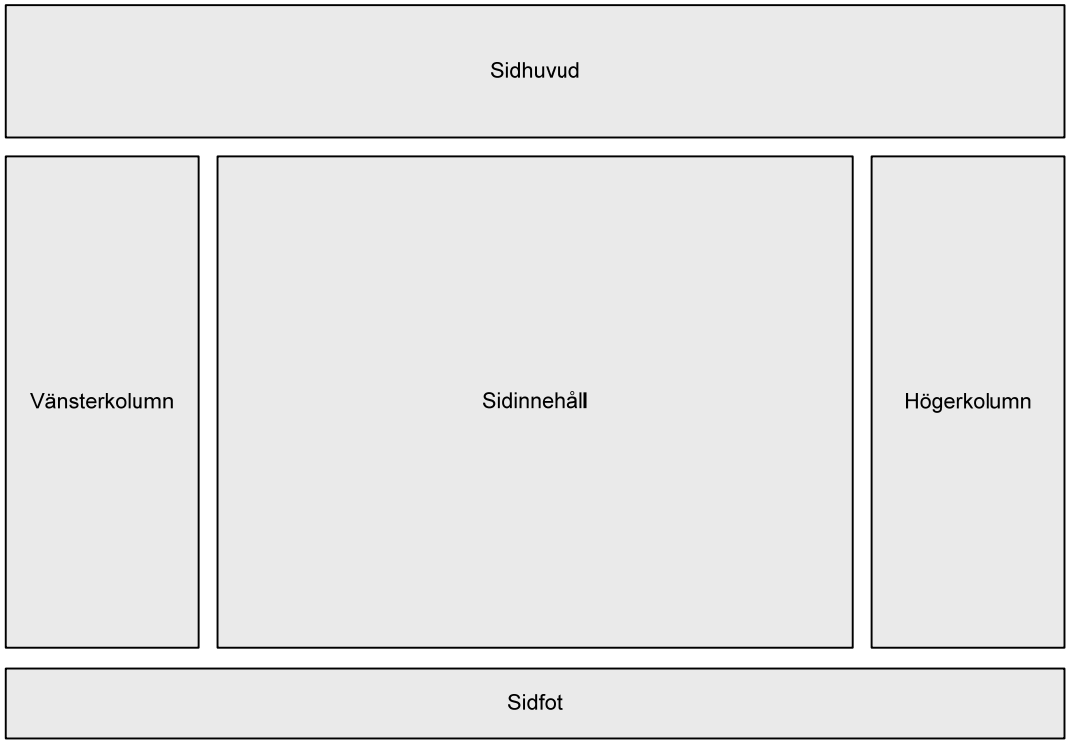
3.1.4.3 Design och Layout

Butiken ska ha en stilren design som tilltalar målgruppen som är mycket trendmedveten. Layouten ska vara intuitiv, konsekvent genomförd och underlätta för besökaren att hitta efterfrågad information. Vissa av Nielsens, Normans samt Shneidermans användbarhetsprinciper kommer användas.

Webbplatsen ska utvecklas i XHTML 1.0 Transitional. För butikens grafiska layout ska uteslutande DIV-element användas. Layout får således inte struktureras med tabeller. XHTML och CSS ska utformas så att grundläggande design och layout enkelt ska kunna förändras genom att enbart byta CSS-mall.

Vidare så ska webbplatsen klara validering enligt W3C-standarden innan den fylls med innehåll.

Layoutens olika segment ska delas upp så att funktionalitet i största möjliga utsträckning ska kunna återanvändas. Figur 8 visar webbplatsens disposition.



Figur 8 Webbplatsens disposition

3.1.5 Funktionella krav

3.1.5.1 Framsida

Butikens framsida ska innehålla produkter som särskilt valts ut ur databasen för att exponeras där. Även senast tillkomna produkter ska visas och produkter som är särskilt populära.

Plats för större bild eller flash-element ska enkelt kunna göras tillänglig för särskilda erbjudanden eller kampanjer.

3.1.5.2 Produktkatalog

Produktkatalogens syfte ska vara att visa produkter i en viss kategori eller av ett valt märke. Även sökresultat ska hanteras av produktkatalogen.

Sidan ska ha en ”paging”-funktion så att inte mer än sex produkter visas på sidan. Det ska finnas möjlighet för användaren att sortera produkter efter märke eller pris. Produkterna ska listas tillsammans med produktnamn, märke och en kort beskrivning. En klickbar mindre bild av produkten ska även visas. Produktens rekommenderade cirkapris ska visas tillsammans med butikens pris.

Användaren ska ha möjlighet att lägga till en produkt i butikens virtuella kundvagn. Har en produkt särskilda attribut som exempelvis färg och storlek så ska användaren kunna välja bland dessa i en rullgardinsmeny.

3.1.5.3 Produktinformation

Produktinformation visas när användaren klickar på en specifik produkt. Informationen ska bestå av en mer ingående beskrivning av produkten tillsammans med en eller flera bilder som ska kunna förstoras.

Användaren ska kunna välja mellan attribut om sådana existerar för vald produkt. Detta kan vara exempelvis storlek och färg. Ändrar användaren attribut så ska det finnas möjlighet att uppdatera bilder och beskrivningar dynamiskt efter valt attribut.

3.1.5.4 Kundvagn

Syftet med kundvagnen är att visa information om valda produkter i form av benämning, märke, attribut, antal, pris och summa. Användaren ska kunna ändra antal eller ta bort produkter.

Användaren ska också kunna ta del av den exakta fraktsumman utan att behöva gå vidare till butikens kassa.

3.1.5.5 Inloggning

Innan användaren kan slutföra sin order genom att gå till butikens virtuella kassa måste inloggning ske. Den e-postadress som användaren registrerat sig med ska användas som användarnamn. Lösenord väljs också av användaren vid registreringen. Har användaren glömt sitt lösenord ska möjligheten finnas att få det skickat till den registrerade e-postadressen.

Är inte användaren registrerad så ska möjligheten att gå vidare till kundregistrering finnas.

3.1.5.6 Kundregistrering

På sidan för kundregistrering ska användaren kunna registrera sig som antingen privatperson eller som företag. Formuläret ska anpassas efter vilket val användaren gör.

För registrering av privatperson ska formulär finnas för förnamn, efternamn, gatuadress, postnummer, postort, land, telefonnummer och mobiltelefonnummer. Användaren ska även kunna välja om leveransadress skiljer sig från fakturaadress.

Även e-postadress och lösenord måste fyllas i för att sedan kunna logga in på sidan.

3.1.5.7 Kassa

I butikens kassa ska användaren kunna slutföra en order. Kassan ska bestå av fyra steg som ska bestå av adressuppgifter, leverans och betalalternativ, bekräftelse och kvitto. När en order har slutförts så ska ett e-postmeddelande skickas till kunden med en orderbekräftelse.

Kassan ska ha möjlighet att hantera betalningar via kort och kunna byggas ut för att ha möjligheten att tillämpa andra betalningsmetoder. Förutom kortbetalning ska postförskott finnas som alternativ.

Även om användaren inte är van vid att handla via Internet så ska han eller hon enkelt kunna förstå hur tillvägagångssättet fungerar. Användaren ska känna sig trygg nog att slutföra en order.

3.1.5.8 Informationssida

Informationssidor kan bestå av information om butiken, kundtjänst eller av kontaktuppgifter. Dessa ska vara dynamiska och ska kunna uppdateras genom butikens administrationsgränssnitt.

3.1.5.9 Konto

En registrerad användare ska under konto kunna ta fram detaljerad orderhistorik om tidigare beställningar. Det ska även vara möjligt att avbeställa ej skickade beställningar eller enbart kontrollera status på en order.

Under konto ska även användaren kunna ändra sina adressuppgifter och lösenord.

3.1.6 Globala funktioner

3.1.6.1 Sökfunktion

En sökfunktion för fritext ska placeras överst i vänsterkolumnen. I framtiden ska även en funktion för avancerad sökning läggas till.

3.1.6.2 Kundvagn

Överst i höger kolumn ska en sammanfattning av kundvagnen placeras så att användaren alltid har uppsikt över vilka produkter som är placerade i kundvagnen. Knappar för att tömma kundvagn och för att gå vidare till kassa ska även finnas tillgängliga här.

3.1.6.3 Inloggning

I sidhuvudet ska det finnas en funktion för att snabbt kunna logga in. När användaren är inloggad visas det här.

3.1.6.4 Produktträd

I vänster kolumn ska tillgängliga kategorier visas tillsammans med en lista med märken som finns representerade i butiken.

Butiken är förberedd för att använda sig av olika avdelningar. När dessa blir aktuella visas de i sidhuvudet.

3.2 Utvecklingsmiljö och verktyg

Det här arbetet bygger uteslutande på Microsofts servertekniker och verktyg. För programmeringen av e-handelsystemet har ASP.Net 2.0 använts som är den senaste tekniken från Microsoft för att bygga dynamiska webbplatser. Jämfört med första versionen av samma teknik så har en rad förbättringar gjorts för att underlätta för programmeraren att öka sin produktivitet. Av dessa förbättringar så har bland annat förmågan att använda *Master Pages* och den nya webbkontrollen (web control) *Wizard* använts. Vad detta är och var det har implementerats behandlas senare i rapporten.

Programmeringsspråk som används tillsammans med ASP.Net kan vara C# eller VB.Net. Vilket som används har ingen betydelse utan är mer en fråga om tidigare preferenser och vad man själv föredrar. Det är även möjligt att använda flera språk samtidigt även om det inte rekommenderas. I det här projektet används enbart C#.

För att utveckla ett projekt av denna storlek underlättar det att använda ett verktyg framtaget för att utveckla webbapplikationer i ASP.Net. Det system som avhandlas i denna rapport har utvecklats med *Visual Studio 2005* och i vissa fall *Visual Web Developer 2005 Express Edition*. Det senare är en enklare version av Visual Studio helt anpassat för webbapplikationer och som dessutom är helt fritt att använda.

Som webbserver för att testa och utvärdera systemet används Microsofts *IIS* (Internet Information Services) men Visual Studio 2005 har även en inbyggd webbserver så att man även utan tillgång till IIS lokalt kan exekvera ASP.Net -applikationer.

Microsoft SQL Server 2005 används som databashanterare och data från databasen hämtas med lagrade procedurer skrivna i T-SQL. Uppbyggnad av databasen och hanteringen av data avhandlas mer i detalj i nästa avsnitt.

3.2.1 Arkitektur

Ett av målen för det här projektet är att göra butiken flexibel nog att kunna användas i olika syften. Även möjligheten att bygga ut systemet efter behov är en viktig faktor.

För att uppnå dessa mål bygger projektet på en så kallad flerskiktsarkitektur där systemets funktionalitet har delats upp i följande tre skikt (lager):

- Datalagret
- Affärslagret
- Presentationslagret

Datalagret sköter i huvudsak lagring av data som i detta fall består av en SQL Server -databas. Även lagrade procedurer omfattas av detta lager. Data hämtas från datalagret till **affärslagret** vars uppgift är att bearbeta detta för att sedan returnera samlingar av data som anropats av **presentationslagret**. Presentationslagret omfattar det användaren ser och integrerar med, det vill säga i detta fall webbplatsens aspx- och ascx-sidor tillsammans med deras code behind.

En viktig del med flerskiktsarkitekturer är att informationsflödet måste röra sig sekventiellt mellan skikten. Presentationslagret får till exempel aldrig kommunicera direkt med datalagret utan att först gå via affärslagret. Bryter man denna regel så riskerar man ett mindre flexibelt system som är svårt att underhålla.

3.3 Databas

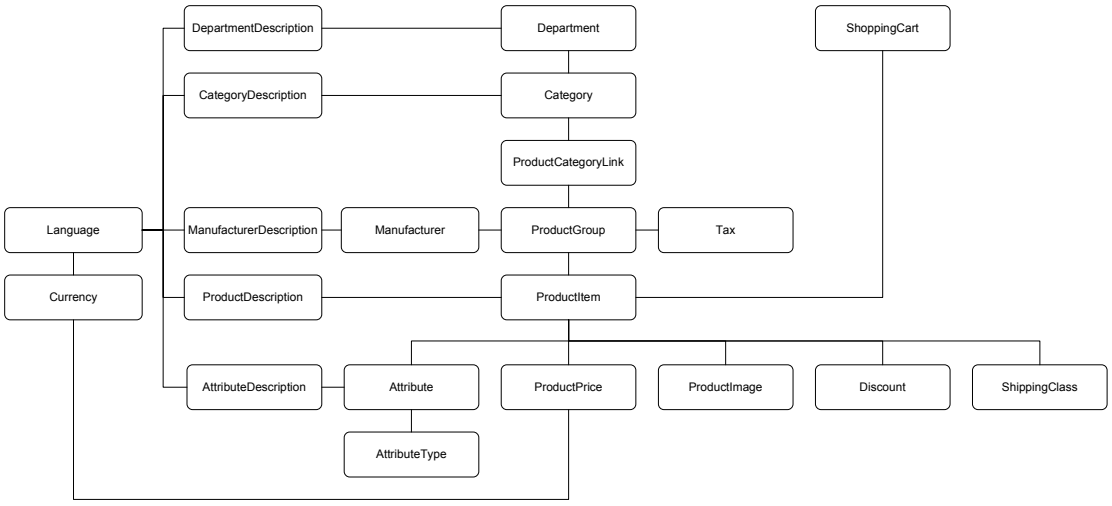
Alla informationssystem behöver någon form av databas för att spara och hantera information. Detta butikssystem är inget undantag. Eftersom en databas utgör en av grundpelarna i ett system är det viktigt att den är välplanerad för att inte i ett senare skede begränsa utvecklingen av övriga delar. Det är också viktigt ur prestandasynpunkt att ha en väl utformad och normaliserad databas med fullgoda relationer.

Eftersom detta projekt bygger på Microsofts servertekniker så föll valet naturligt på att använda Microsoft SQL Server 2005 som databashanterare. Den version som använts i utvecklingssyfte har tillägget *Express Edition* vilket innebär att den är gratis men har vissa begränsningar i antal samtida användare. All information om produkter, kategorier, kundvagn, beställningar och kunder lagras och hanteras av SQL Server.

Databasen har byggts upp i *SQL Server Express Manager* som är ett fritt program att ladda hem från Microsoft. Även Visual Studio 2005 har funktionalitet som har använts till uppbyggnad och administration.

För att enklare kunna beskriva databasen så har den på följande sidor delats upp i två delar varav den ena är produktkatalog och den andra orderhantering. För ett detaljerat schema över hela databasen hänvisas till bilaga 1.

3.3.1 Produktkatalog



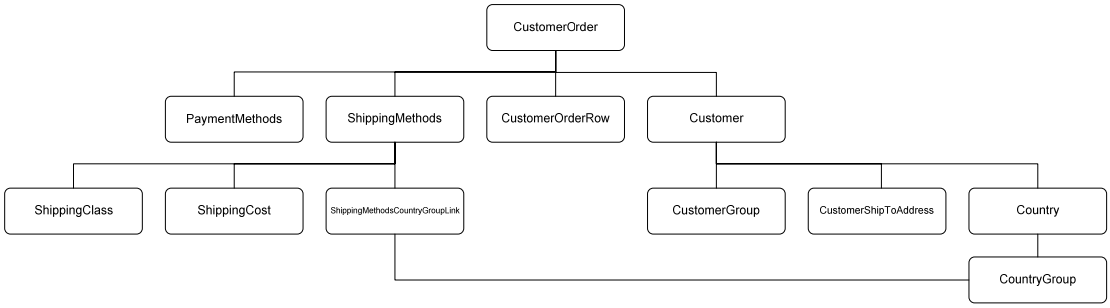
Figur 9 Diagram över tabeller för produktkatalog i databasen

Produktkatalogen, se Figur 9, utgörs av den samling tabeller som beskriver en produkts hierarki och information. En produkt (*ProductItem*) definieras av den fysiska varan, tillsammans med dess attribut, en kund kan köpa. Till exempel en grön tröja i storlek L. Finns samma tröja i flera storlekar så tillhör den en produktgrupp (*ProductGroup*) där en produkt är förvald, det vill säga den produkt som visas i en kategori (*Category*). En produktgrupp kan således innehålla många produkter med olika attribut. I och med detta finns en risk för redundans men å andra sidan är det möjligt att ange individuella beskrivningar, bilder, prissättning och lagerstatus för varje enskild produkt.

Varje produktgrupp tillhör en eller flera kategorier som är rekursiv vilket innebär att en kategori kan innehålla flera underkategorier. Varje kategori tillhör i sin tur en avdelning (*Department*).

Databasen är väl förberedd för att i framtiden ha en möjlighet att göra butiken multinationell. Detta åstadkoms genom att all data som presenteras på sidan och varierar beroende på språk, ligger i tabeller avsedda för detta. För en produkt ligger produktbeskrivningar i tabellen *ProductDescription*, för kategorier i *CategoryDescription* och så vidare.

3.3.2 Orderhantering



Figur 10 Orderhantering i form av tabeller i databasen

Figur 10 visar hur orderhanteringen ser ut i form av tabeller. Tabellen *CustomerOrder* är huvudtabellen för en order. Här sparas information om orderdatum och status samt de främmande nycklar som är relevanta för betalning (*PaymentMethods*), frakt (*ShippingMethods*) och kund (*Customer*). Orderraderna sparas i en tabell för sig (*CustomerOrderRow*) där en produkts id, artikelnummer, namn, pris etcetera sparas för att det ska vara möjligt att få tillgång till orderhistorik även om en produkt avlägsnas från databasen.

En kund kan tillhöra olika grupper (*CustomerGroup*) för till exempel återförsäljare. Kundens fakturaadress sparas i *Customer* men kunden kan även ha en eller flera leveransadresser (*CustomerShipToAdress*). De länder en kund kan befinna sig i ligger i tabellen *Country*.

Länderna är även uppdelade i grupper som till exempel Norden, Europa och Internationellt för att underlätta prissättning för frakt runt om i världen.

3.3.3 Lagrade Procedurer

Istället för att skriva T-SQL direkt i applikationens källkod skrivs detta i *Lagrade Procedurer* (Stored Procedures) som sparas internt i databasen. Dessa lagrade procedurer anropas av webbapplikationen och kan antingen hämta, manipulera eller spara data i databasen.

Kodexempel 3 visar ett exempel på en enkel procedur som används i butiken samt hur denna anropas av applikationen.

GetCategoriesInDepartment

```
CREATE PROCEDURE GetCategoriesInDepartment
(
	@DepartmentID int = 1,
	@LanguageID int = 1
)
AS

SELECT	Category.CategoryID,
	CategoryDescription.CategoryName

FROM	Category INNER JOIN
	CategoryDescription ON
	Category.CategoryID = CategoryDescription.CategoryID

WHERE	(Category.DepartmentID = @DepartmentID) AND
	(CategoryDescription.LanguageID = @LanguageID) AND
	(Category.CategoryPublish = 1) AND
	(Category.CategoryParentID = 0)
```

Kodexempel 3 GetCategoriesInDepartment

Vad detta gör är att hämta alla huvudkategorier, tillsammans med dess ID, i en viss avdelning förutsatt att kategorin är publicerad. *@DepartmentID* och *@LanguageID* är så kallade parametrar som skickas med vid anropet. I detta fall behöver vi veta vilken avdelning som är vald och vilket språk besökaren använder. Är inte dessa parametrar specificerade så väljs i det här fallet automatiskt värdet 1.

När man skapar en lagrad procedur börjar man alltid med CREATE PROCEDURE. När denna sedan är sparad och man vill ändra i den skrivs ALTER PROCEDURE. Visual Studio sköter dock detta automatiskt. I Visual Studio kan man även testköra en procedur utan att först använda den i applikationen. Parametrar anges då manuellt innan exekvering.

ProductCatalogAccess.cs

```
public static DataTable GetCategoriesInDepartment(string departmentId, string
languageId)
{
        DbCommand comm = GenericDataAccess.CreateCommand();
        comm.CommandText = "GetProductsInCategory";

        DbParameter param = comm.CreateParameter();
        param.ParameterName = "@DepartmentID";
        param.Value = departmentId;
        param.DbType = DbType.Int32;
        comm.Parameters.Add(param);

        param = comm.CreateParameter();
        param.ParameterName = "@LanguageID";
        param.Value = languageId;
        param.DbType = DbType.Int32;
        comm.Parameters.Add(param);

        DataTable dt = GenericDataAccess.ExecuteSelectCommand(comm);

        return dt;

}
```

Kodexempel 4 ProductCatalogAccess.cs

I kodexempel 4 används klasserna i GenericDataAccess för att skapa kopplingen mot databasen och göra anropet till proceduren tillsammans med de specificerade parametrarna. När det har gjorts fylls en DataTable med resultatet och returneras.

Kodexempel 5 och 6 har inte med lagrade procedurer att göra men kan vara intressant i sammanhanget eftersom det beskriver hur klassen ovan anropas av presentationslagret och hur den DataTable som använts kopplas till en Repeater.

```
DataTable dtCategories = ProductCatalogAccess.GetCategoriesInDepartment();

repCategory.DataSource = dtCategories;
repCategory.DataBind();
```

Kodexempel 5 CategoryList.ascx.cs

```
<ul id="menuCategories">
    <asp:Repeater ID="repCategory" runat="server">
        <ItemTemplate>
            <li><a href='Catalog.aspx?CategoryID=<%# Eval("Id") %>'>
            <%# Eval("Name") %></a></li>
        </ItemTemplate>
    </asp:Repeater>
</ul>
```

Kodexempel 6 CategoryList.ascx

3.4 Integrering av betallösning

Ett av projektets huvudmål är att integrera en betallösning i e-handelssystemet för att kunna erbjuda möjligheten till kortbetalningar. Den tjänst som har valts för att kunna realisera detta är Samports API-tjänst. Anledningen till att valet föll på denna tjänst är att den ger en kompromiss mellan kostnadseffektivitet och flexibilitet.

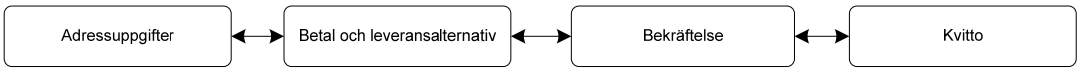
Utöver ovan nämnda betalväxel krävs ett SSL-certifikat för att säkerställa kryptering av data som skickas mellan applikationen och Samports API. I utvecklingssyfte skaffades ett 30 dagars kostnadsfritt prov av ett certifikat från företaget Comodo via www.instantssl.com.

Slutligen behövs ett avtal om inlösen med en bank om betallösningen ska fungera i praktiken. I denna rapport behandlas enbart utvecklingsstadiet där korttransaktioner endast simuleras. Därför behövs ej detta avtal förrän butiken ska användas i skarpt syfte.

Kommunikationen med Samports API sker via anrop direkt i applikationens kod. All information som krävs för att auktorisera transaktionen samlas ihop och skickas till en WebService hos Samport. Informationen kontrolleras och en svarskod skickas tillbaka till applikationen beroende på om köpet är godkänt eller inte.

Integreringen av betallösningen görs i butikens virtuella kassa tillsammans med klassen *CreditCardProcessing.* Bilaga 2 visar ett schema över kortbetalningsflödet.

Kassan består av fyra steg (om man inte räknar med inloggning) som består av adressuppgifter, leverans och betalalternativ, bekräftelse och kvitto. Se Figur 11.



Figur 11 Kassaflöde

Under **adressuppgifter** visas användarens faktura- och leveransadress för att bekräfta att uppgifterna som är lagrade i databasen är korrekta. Om användaren vill ändra registrerad adress så kan det göras här. Användaren har även möjlighet att lägga till en godsnotering.

Nästa steg är **betal- och leveransalternativ** där användaren väljer hur han/hon vill betala ordern och hur den ska levereras. Väljs kortbetalning visas ett formulär där kortuppgifter fylls i.

Vilka leveransalternativ som finns tillgängliga, och till vilket pris, beror på vilket land användaren befinner sig i och vilken fraktklass produkterna i kundvagnen har. Priset anpassas till den produkt som har högst klass.

Som tredje steg visas en **bekräftelse** där all information användaren har valt och fyllt i, tillsammans med kundvagnens innehåll. Användaren bekräftar sin order vid detta steg.

Vid fjärde och sista steget har ordern sparats i databasen och användaren får ett **kvitto** på att allt är klart. Ett kvitto skickas samtidigt till användarens e-postadress.

Dessa fyra steg hanteras av en kontroll som heter Wizard vilken har tillkommit i ASP.Net 2.0. Syftet med denna kontroll är att skapa gränssnitt där inmatning av data delas upp i flera steg och där Wizard-kontrollen behåller samma viewstate genom hela processen så att all inmatning enkelt kan hämtas fram och tillbaka mellan stegen.

Kodexempel 7 visar en förenklad version den Wizard som används till butikens kassa. När en användare byter steg triggas händelsen *Wizard1_ActiveStepChanged* vilket visas i kodexempel 8.

```
<asp:Wizard ID="wizCheckout" runat="server"
    OnActiveStepChanged="Wizard1_ActiveStepChanged" DisplaySideBar="false">

    <WizardSteps>
        <asp:WizardStep ID="stepAddress" runat="server" Title="Step 1">

            <!-- Adressuppgifter -->

        </asp:WizardStep>
        <asp:WizardStep ID="stepPayment" runat="server" Title="Step 2">

            <!-- Betal och leveransalternativ -->

        </asp:WizardStep>
        <asp:WizardStep ID="stepConfirm" runat="server" Title="Step 3">

            <!-- Bekräfta -->

        </asp:WizardStep>
        <asp:WizardStep ID="stepReciept" runat="server" Title="Step 4">

            <!-- Kvitto -->

        </asp:WizardStep>
    </WizardSteps>

</asp:Wizard>
```

Kodexempel 7 Något förenklad version av den Wizard som används till Kassan

```
protected void Wizard1_ActiveStepChanged(object sender, EventArgs e)
{
     string stepID = Wizard1.ActiveStep.ID;

     if (stepID == "stepAddress")
     {
         ltrCheckoutStep.Text = "Adress";
     }


}
```

Kodexempel 8 Händelse vid stegbyte

Väljs kortbetalning vid andra steget så får användaren uppge kortnummer, giltighetsdatum och cvv-kod. När användaren går vidare till nästa steg görs en snabb validering av kortnumret genom metoden ValidateCreditCard i klassen CreditCardProcessing, se kodexempel 9.

```
if (stepID == "stepConfirm" && rdoPaymentCard.Checked)
{
     string creditCardNo = txtCardNo.Text;

     if (!CreditCardProcessing.ValidateCreditCard(creditCardNo))
     {
         lblError.Visible = true;
         lblError.Text = "Felaktigt kortnummer.";

         Wizard1.MoveTo(this.stepPayment);
     }
}
```

Kodexempel 9 Anrop av ValidateCreditCard

I koden ovan anropas metoden ValidateCreditCard om kortbetalning har valts av användaren. Retuneras ett falskt värde så visas ett felmeddelande och användaren uppmanas till att fylla i ett nytt nummer.

```
public static bool ValidateCreditCard(string cardNo)
{
    Samport.SamportWSAPI card = new Samport.SamportWSAPI();

    return card.ValidateCardNum(cardNo);
}
```

Kodexempel 10 Anrop av Samports Webservice

I kodexempel 10 visas Metoden ValidateCreditCard som i sin tur anropar metoden *ValidateCardNum* via Samports WebService med kortnumret. Valideras kortet korrekt retuneras *true*. Är kortnumret felaktigt retuneras *false*.

När kortet har validerats registreras ordern i databasen men förblir obekräftad tills användaren själv bekräftar sin order i nästa steg i kassan. Anledningen till att ordern registreras redan innan ordern har lagts är för att ett ordernummer krävs vid initieringen av korttransaktionen. För att bli tilldelad ett ordernummer måste ordern sparas i databasen. I praktiken utgör inte detta något större problem. Ordern registreras först med statusen obekräftad och ändras när ordern väl läggs. Detta kan vara ett sätt att analysera hur många ”övergivna kundvagnar” butiken får. Är antalet oroväckande många betyder det oftast att något gör användaren så pass osäker att han eller hon inte vågar slutföra beställningen.

Nästa steg i kortbetalningsprocessen är att initiera ett auktoriseringsanrop vilket visas med kodexempel 11 och 12. Eftersom butiken ska ha möjlighet att utnyttja 3D Secure så används metoden *D3DSecure_Initiate* som är avsedd för detta. Metoden kräver följande argument:

- TellusPayID
- OrderNo
- Amount
- CardData
- CurrencyCode
- ResponseURL

TellusPayID är kundnumret hos Samport.

OrderNo är ett unikt nummer på aktuell beställning. Detta nummer sparas hos Samport för att man ska kunna hantera transaktioner i deras back-office-system.

Amount är den totala summan som ska reserveras på användarens konto. Summan anges i lägsta möjliga valör.

CardData består av kortnumret tillsammans med giltighetsdatum i formen XXXXXXXXXXXXXXXX=YYMM.

CurrencyCode är valutakod för den valuta transaktionen ska utföras med. Svenska kronan har koden 752.

ResponseURL är webbadressen som Samport skickar användaren efter 3DSecure verifieringen. Adressen för det här projektet i utvecklingsskedet är ”http://dev.thirdfloor.biz/Internetbutik/checkout.aspx?order=1234&SecureID=” . I slutet av strängen läggs ett id automatiskt till.

```
DataSet dsInitiate = CreditCardProcessing.Initiate3DSecure(orderNo, amount,
cardData, cvv2);
```

Kodexempel 11 Anrop av Initiate3DSecure.

```
public static DataSet Initiate3DSecure(string orderNo, string amount, string
cardData, string cvv2)
{
    string tellusPayId = "xxxxxxxx";
    string currencyCode = "752";
    string responseUrl =
    "https://dev.thirdfloor.biz/Internetbutik/checkout.aspx?OrderNo=" + orderNo +
    "&CVV2=" + cvv2 + "&SecureID=";

    //-------------------------------------------------------------------------
    //Base64 encode url

    byte[] url = System.Text.Encoding.UTF8.GetBytes(responseUrl);
    responseUrl = Convert.ToBase64String(url);

    //-------------------------------------------------------------------------
    //Connect to Samport WebService and recieve dataset with return data.

    Samport.SamportWSAPI card = new Samport.SamportWSAPI();

    DataSet ds = card.D3DSecure_Initiate(tellusPayId, orderNo, amount, cardData,
    currencyCode, responseUrl);

    //-------------------------------------------------------------------------
    //Return DataSet

    return ds;

    //-------------------------------------------------------------------------
}
```

Kodexempel 12 Initiate3DSecure och anrop av D3DSecure_Initiate.

D3DSecure_Initiate returnerar sedan ett DataSet med två parametrar. *SecureID* och *3DSecURL* där den förstnämnda är ett unikt 3D Secure initieringsid. 3DSecURL består av en webbadress som användaren ska skickas till för verifiering. Har inte kortet stöd för 3D Secure så retuneras strängen ”NO”. I det fallet behövs inte verifieringen göras och nästa steg i betalprocessen kan utföras istället.

När transaktionen har initierats och stöd för 3D Secure finns är nästa steg att skicka användaren till en sida hos kortutfärdaren (VISA eller MasterCard) där han eller hon uppmanas till att fylla i sitt lösenord för verifiering. När verifieringen slutförts skickas användaren tillbaka till butiken och auktoriseringen av kortet kan påbörjas.

Auktorisering av en transaktion görs med metoden D3DSecure_AuthorizeCard (kodexempel 13 och 14) som vill ha argumenten:

- TellusPayID
- TerminalID

- CVV2
- SecureID
- ProcessingCode
- DirectCapture
- SecretKey
- IP

TerminalID är ett unikt nummer på den virtuella terminalen som används. Jämför med den fysiska terminal man ofta ser i butiker.

CVV2 är en säkerhetskod som består av tre eller fyra siffror och ofta finns på kortets baksida.

SecureID används för att indentifiera uppgifterna från initieringen.

ProcessingCode indikerar vilken typ av transaktion som ska utföras. ”00nnnn” är koden för ett köp.

DirectCapture avgör om summan ska dras från kortet direkt eller bara reserveras. I butiken reserveras endast summan och dras sedan när ordern har skickats till kund.

SecretKey är en krypteringsnyckel som fås av Samport.

IP är användarens ip-nummer som sparas ifall bedrägerier skulle uppstå.

```
DataSet dsAuthorize = CreditCardProcessing.Authorize3DSecure(secureID, cvv2);
```

Kodexempel 13 Anrop av Authorize3DSecure.

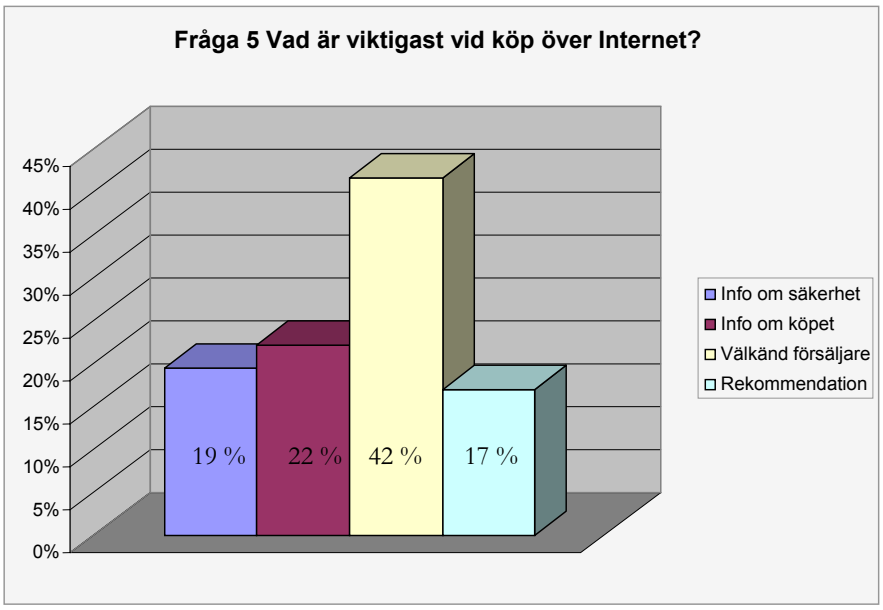
```
public static DataSet Authorize3DSecure(string secureID, string cvv2)
{
    string tellusPayId = "xxxxxxxx";

    string terminalId = "xxxxxxxxxx";

    string processingCode = "002000";

    bool directCapture = false;

    string ip =
    HttpContext.Current.Request.ServerVariables["REMOTE_ADDR"].ToString();

    //-------------------------------------------------------------------------
    //Connect to Samport WebService and recieve dataset with return data.

    Samport.SamportWSAPI card = new Samport.SamportWSAPI();

    DataSet ds = card.D3DSecure_AuthorizeCard(tellusPayId, terminalId, cvv2,
    secureID, processingCode, directCapture, ip);

    //-------------------------------------------------------------------------
    //Return dataset with response code

    return ds;

    //-------------------------------------------------------------------------
}
```

Kodexempel 14 Authorize3DSecure och anrop av D3DSecure_AuthorizeCard

D3DSecure_AuthorizeCard returnerar en ”response code” beroende på om transaktionen godkänns eller inte. ”00” indikerar en godkänd transaktion och ”T0” en godkänd testtransaktion. I övriga fall retuneras felkoder för olika typer av fel och användaren uppmanas till att antingen fylla i ett nytt kortnummer eller välja ett annat betalningsalternativ. När transaktionen har auktoriserats bekräftas ordern i databasen och ett e-postmeddelande skickas till användaren med en orderbekräftelse.

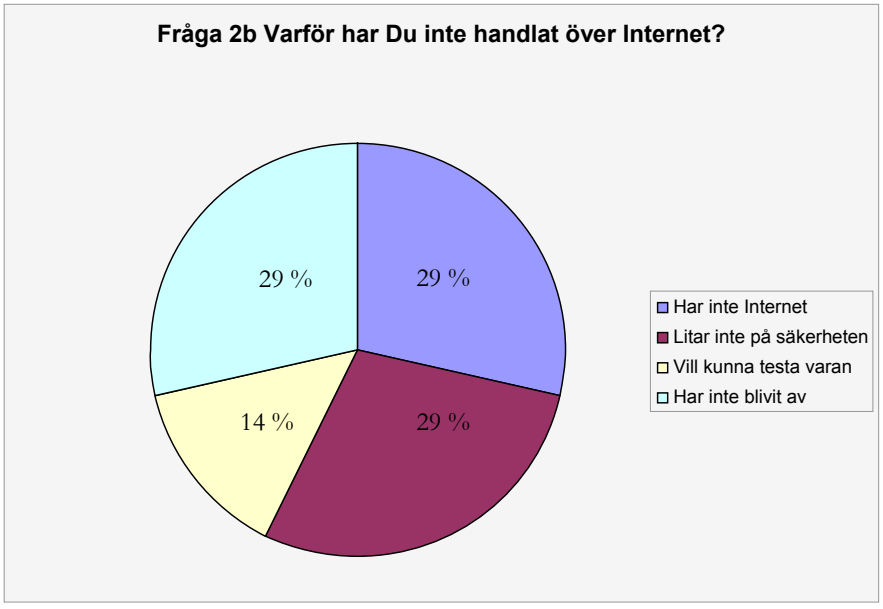
3.5 Undersökning

För att undersöka hur människor i allmänhet uppfattar betalningar på Internet så har en enkät med frågor angående detta, tagits fram, se bilaga 3. Denna enkät genomfördes av 56 personer i alla åldrar mellan 18 och 60+ och av båda könen. Samtliga svarande kommer från samhällen och städer från östra delen av Västra Götalands län, vilket kan ha viss effekt på resultatet.



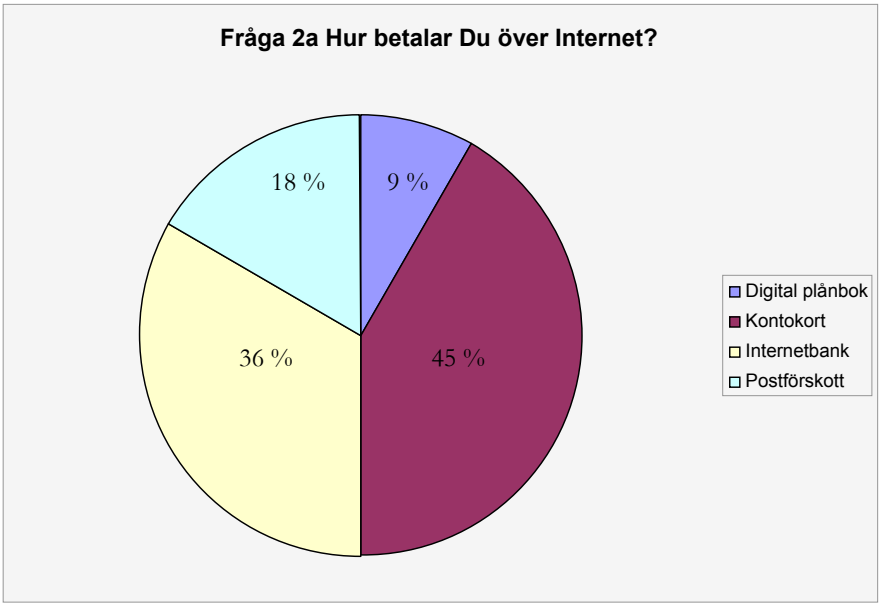
Figur 12 Diagram över vad som är viktigast vid köp över Internet

De flesta personer som gjorde vår enkät ansåg att butiken skulle vara välkänd för att de skulle våga köpa något från den, se Figur 12. Alltså, att få ett gott rykte och många nöjda kunder kan vara nog så viktigt som att visa vilken säkerhet man har på sidan. Ju fler som känner till företaget som säljer produkterna, desto kändare blir det, vilket gör att folk som ska handla där känner sig säkrare.



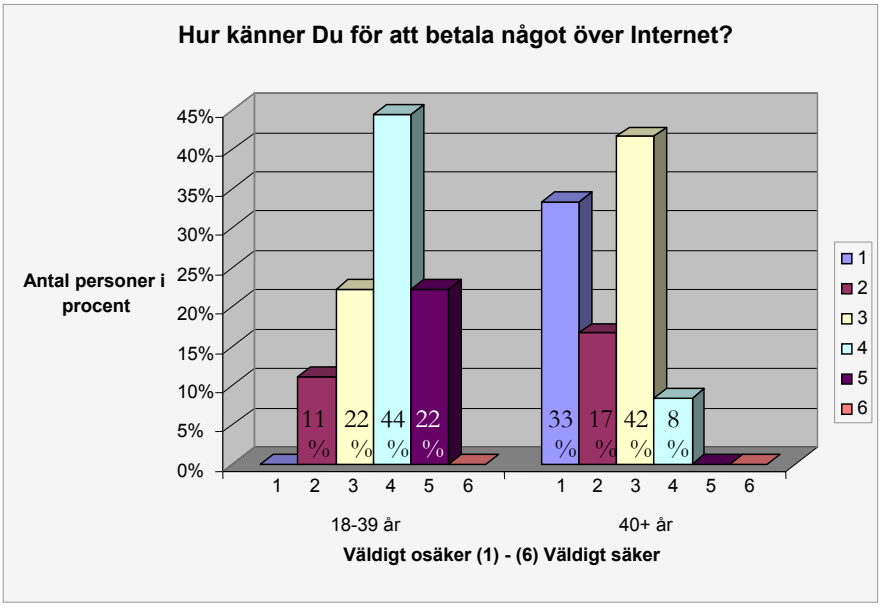
Figur 13 Diagram över varför folk inte har handlat över Internet

Figur 13 visar de personer som svarade på vår enkät och som inte handlat över Internet. Den minsta orsaken var att de ville kunna testa varan i butik. Flera stycken hade inte Internet och för lika många hade det helt enkelt inte blivit av. Dessutom var det samma antal som hade svarat att de inte litade på säkerheten. För att få fler folk att våga ta steget och genomföra ett köp över Internet är således säkerheten väldigt viktig.



Figur 14 Diagram över hur folk betalar över Internet

Det vanligaste sättet att betala något över Internet var med kontokort av dem som svarade på enkäten, tätt följt av Internetbank, se Figur 14. Att betala med hjälp av en digital plånbok var inte alls lika populärt som de två tidigare alternativen. Antingen anser de tillfrågade att kontokort och Internetbank att det är säkrare än digital plånbok, eller så tycker dem att det är enklare att förstå och använda sig av. Hur som helst är det viktigast att alternativen kontokort och Internetbank finns, där den största kundkretsen finns. Men både postförskott och digital plånbok är bra komplement för att kunna tillfredsställa de flesta eventuella kunder.



Figur 15 Diagram över vad folk känner för att betala över Internet

Uppdelat efter ålder vid frågan om hur säkra folk känner sig för att betala något över Internet så syns det ganska bra att unga har en lite säkrare bild av Internethandel, se Figur 15. Flera i den äldre delen kände sig väldigt osäkra över att handla på Internet, medan det i den yngre gruppen var flest personer som tyckte att det kändes mer säkert än osäkert att göra ett köp online. Sammanfattningsvis känner de yngre generationerna sig säkrare med att betala över Internet än de äldre. Orsaken till detta kan vara det att den yngre generationen mer eller mindre har vuxit upp med Internethandel.

Nedan följer ett urval av kommentarer på de två sista frågorna i enkäten som var av kommentarkaraktär. Frågorna löd: ”Vad tycker Du webbutiker ska göra för att fler ska våga/vilja köpa något över Internet?” och ”Övriga kommentarer angående betalningar över Internet.” På den första av de här frågorna var några av svaren följande:

- ”Fortsätta jobba med enkel information.”
- ”Bytesgaranti, fri/billigare frakt.”
- ”Säkerheten säkrare, kontroll av oseriösa handlare bör ökas.”
- ”Enkelt och lättbegripligt att handla.”
- ”Utöka säkerheten i köpet.”

På övriga kommentarer var svaren dessa:

- ”Fler butiker som länkar direkt till Internetbanker.”

- ”Är det ett välbekant företag så känns det tryggare att handla över Internet.”

- ”Att det blir så enkelt att köpa eller sälja varor.”

Utifrån de här kommentarerna syns det att enkelhet är något som anses viktigt när det gäller att handla över Internet. Även säkerheten tas upp en del och att ha en länk till Internetbanker på butikens sida var något som önskades.

På frågan om de tillfrågade hade handlat över Internet förr svarade 64 % att de hade gjort det, och resterande 36 % att de inte hade gjort några köp över Internet. Däremot var det 79 % av de svarande som trodde att de skulle göra ett köp över Internet det närmaste året. Enbart 21 % trodde inte att de kommer att göra det. Alltså kan man se att i denna grupp som tillfrågades, kommer troligen köp online att öka även idag.

4 Resultat

4.1 Frågeställningar och mål

De mål som Internetaffären skulle uppnå för företaget var att den på ett enkelt och stilrent sätt skulle kunna locka till sig kunder. Den skulle vara lätt att använda så att kunden skulle känna igen sig även fast det var en helt ny affär som de inte använt innan. Designen skulle vara enkel och stilren så den kändes exklusiv och nyskapande. Affären skulle även kunna sälja andra varor än kläder.

Målet sett ur vårt perspektiv var att uppfylla företagets mål, få mer kunskap om objektorienterad webbutveckling och säkerhet på Internet. De frågeställningar som sattes upp i början av detta arbete tog upp väldigt viktiga delar av hur en utvecklare och försäljare måste tänka för att en kund vill handla över Internet.

4.2 Avstämning

4.2.1 Allmänt

Examensarbetet har resulterat i en fungerande Internetaffär vilket förhoppningsvis kommer möjliggöra en fortsatt utveckling för företaget.

Rapporten tar upp hur ett e-handelssystem utvecklas, detta genom Kapitel 2 vilket bl.a. behandlar databasteori, användbarhet och olika betaltekniker. Den beskriver även hur god säkerhet uppnås, bl.a. genom olika säkerhetstekniker, hur certifikat fås och hur lösenord kan skyddas.

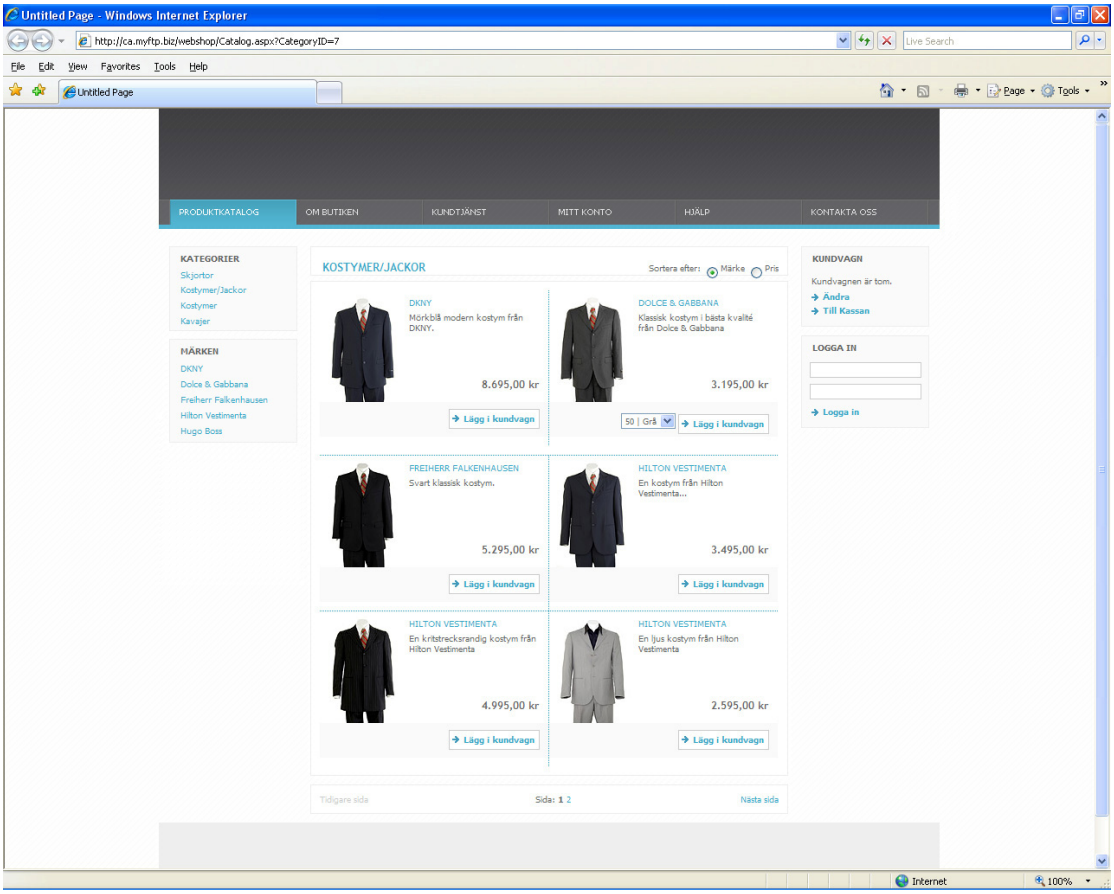
Olika betaltekniker beskrivs, då de är ytterst viktiga i en e-handelslösning. Exempel på bl.a. betalväxlar och digitala plånböcker ingår med detaljerad information samt prisexempel på de mest intressanta tjänster de annonserar.

Rapporten behandlar även om hur kunder ska känna sig trygga och vilja köpa produkter. Exempelvis så tas olika regler för användbarhet upp som bör följas. Vi i gruppen har även skapat en enkät där personer har fått säga sin åsikt om Internethandel och säkerheten på Internet. Senare i detta kapitel skriver vi även om upplevd och faktisk trygghet vilket innebär att även om en Internetaffär ser säker ut så behöver den inte vara det, och vice versa.

Tyngdpunkten i det praktiska arbetet var att skapa en databas som skulle kunna användas för alla sorters produkter, vilket gjorde att databasen fick ett stort antal tabeller och relationer. Vi implementerade även en funktion som gör att kunder ska kunna välja språk på sidan, denna har vi inte använt oss av men funktionen finns i databasen för framtida användning.

Genom att använda en enkel och stilren design så ska kunden känna sig säker att använda Internetaffären. I kapitel 3 beskrivs arbetsgången med affären, samt kravspecifikationen redovisas. Betaltekniken och säkerheten har haft stor prioritet när vi jobbade med affären, vilket även visat sig i rapporten.

4.2.2 Design och layout



Figur 16 En kategorisida i butiken

Designen på affären har utvecklats med tanke på vissa aspekter, då målgruppen anses ha tillgång till skärmar med en upplösning av minst 1024 x 768. Så av layoutmässiga och estetiska skäl har sidans bredd ej överstigit dessa mått samtidigt som bredden är statisk och centreras i webbläsarfönstret.

Layouten är konsekvent utförd och underlättar för kunden att hitta produkter i affären. Vi använde vissa punkter av Nielsens, Normans samt Shneidermans användbarhetsprinciper vid designen av affären, några av dessa var:

- *Konsistens*
  Språk och struktur i systemet används på samma sätt över hela systemet då CSS-mallar, Masterpage och Web User Controls möjliggör att samma design visas på alla sidor i systemet.

- *Gör uppgiftens struktur enkel*
  De saker en kund kan utföra i systemet är enkla och kräver ingen större ansträngning.

- *Reducera belastningen på korttidsminnet*
  Då det finns begränsningar i det mänskliga korttidsminnet så passar vår enkla men stilrena design idealiskt.

4.2.3 Kravspecifikation

I kravspecifikationen beskrivs punkter som behövs för tekniska krav, säkerhet, icke funktionella krav, funktionella krav och globala funktioner. Vid en jämförelse mellan resultatet och kravspecifikationen ser vi att resultatet skiljer sig något. En del mindre saker har ej ännu hunnits med, däribland en funktionell sökfunktion. Dock har inte många undantag från kravspecifikationen gjorts vilket visar att resultat blivit som det var tänkt.

4.2.4 Svar på frågeställningarna

- *Hur utvecklas ett e-handelssystem?*
  Först och främst måste en databas skapas som kan innehålla all information som Internetaffären behöver. Efter detta kan affären skapas med hjälp av ASP.NET. När detta är gjort och ett informationsflöde är kopplat mellan databasen och sidan så måste en betalningslösning implementeras.

- *Hur uppnås tillräcklig säkerhet?*
  Efter att ha läst böcker samtidigt som vi har egna erfarenheter om detta ämne har vi en god bild över hur säkerhet uppnås, både faktisk och upplevd säkerhet. En enkät utfördes även där ett antal personer fick säga sin mening om säkerhet och betalningar över Internet.

  Förutom användande av en betalväxel krävs också ett SSL-certifikat för att säkerställa kryptering av data som skickas mellan applikationen och betalväxel. För att välja en tillräckligt säker betalväxel som fyller det behov som företaget har är det smart att först undersöka vad andra företag som använt samma tjänst ansett om det.

  Det är även bra att tänka på att utveckla systemet enligt PCI-standarden vilket innebär bland annat att inga kortuppgifter lagras i butikens egna databas. Dessa uppgifter omhändertas istället av företaget som tillhandahåller betaltjänsten.

Betaltjänster via Internet anses vara relativt säkra idag och många betalväxlar och kortutfärdare erbjuder tjänster som till exempel 3DSecure för att öka säkerheten ytterligare.

För att skydda butikens egna databas mot angrepp genom SQL-injections är en lösning att använda parameterbaserade lagrade procedurer. Det är också viktigt att kryptera känslig information som till exempel lösenord i databasen.

- *Hur fungerar betaltransaktioner via Internet tekniskt och praktiskt?*
  Denna rapport innehåller information om de mest kända funktioner och säkerhetslösningar, samtidigt som vi själva har implementerat en betallösning med SSL och 3D Secure.

- *Hur skapas en trygghet för e-handelsplatsens besökare?*
  En sida måste visa, redan från första gången en kund kommer till sidan, att den är seriös, och att kunden kan känna sig säker att handla där. Vi använde oss av användbarhetsprinciper samt en stilren design som tilltalar målgruppen.

  Resultaten från vår enkät visade att personer vill ha information om hur saker och ting går till på sidan, men allt tekniskt som sker behöver nödvändigtvis inte redovisas. Enkäten visade också att sidan gärna ska vara enkel och lätt att förstå för att kunder ska känna sig trygga med att betala över Internet.

4.3 Upplevd och faktisk trygghet

Trygghet kan delas in i faktisk trygghet och upplevd trygghet. I en Internetbutik kan faktisk trygghet yttra sig genom den underliggande säkerheten i kod och kryptering, alltså den verkliga säkerheten. Upplevd trygghet är hur trygg kunden känner sig vid ett köp från hemsidan i detta fall. Dessa två kan skilja sig en del. Bara för att den faktiska tryggheten är bra, behöver inte kunden känna sig trygg, om det exempelvis finns dålig information. Likadant kan den upplevda säkerheten vara hög även om den faktiska tryggheten inte är det. I ett fall då butiken uppger sig vara väldigt säker men inte har belägg för det är ett exempel.

Om man då har den rätta tekniken och lyckats skapa en säker butik med hög faktisk trygghet, hur lyckas man då få kunden att förstå att hemsidan är så säker som den är? Att informera är ett måste, om inte kunden kan få reda på information om hur säkert det är, då finns det heller inget som säger att det är en trygg butik.

Enligt dem som svarade på vår enkät var det en välkänd försäljare som gav den största tryggheten. Att då visa på butikens sida på något sätt vad det är för butik, vem som äger den, om det finns någon fysisk affär o.s.v. är en bra början. Vidare på enkäten tyckte de tillfrågade att det var lite viktigare med att få information om hur köpet går till än att få veta om säkerheten bakom. Kanske kan det betyda att folk hellre har enkel text över hur man ska bära sig åt på sidan än tekniskt språk om säkerhet som de flesta ändå inte förstår.

Något som många tycker är ett stort problem när det gäller att uppleva en webbutik trygg är att lämna ut sitt kontonummer. Men att ge ut sitt kontokortsnummer på Internet borde inte vara något problem vad det gäller säkerheten, lika lite som när det används manuellt i en vanlig affär.

5 Slutsats och diskussion

5.1 Examensarbete

Syftet med detta examensarbete var att skapa en fungerande e-handelslösning åt BRND Clothing & Accessories, som ska sälja exklusiva kläder över Internet. Vi ville öka vår kunskap inom objektorienterad webbutveckling och säkerhet på Internet. Vi ville även utveckla en e-handelslösning som var enkel att anpassa för andra områden än kläder.

Att utveckla en så pass omfattande Internetaffär som vi har gjort har visat sig vara tidskrävande men det har även varit ett väldigt intressant arbete.

Tyngdpunkten i det praktiska arbetet har legat på att skapa en databas vilken möjliggör enkel anpassning för andra områden än kläder. Då målgruppen tros vara trend- och stilmedvetna så är butikens design enkel och stilren. För att göra sidan enkel att använda har vissa av Nielsens, Normans samt Shneidermans användbarhetsprinciper använts.

Frågeställningarna som behandlades i början av denna rapport tycker vi har besvarats. Detta visar sig genom denna rapport som innehåller teori, genomförande och resultat om detta arbete. Målen med uppdraget får anses uppnådda och det praktiska arbetet som utförts är vi nöjda med.

Det har varit intressant att se att de kunskaper vi fått under utbildningen har kunnat användas i praktiska fall. Examensarbetet har även gett oss mycket erfarenhet som vi förhoppningsvis har användning av i arbetslivet.

5.2 Framtida arbete

Databasen är anpassad för att kunna användas av många olika produkter, vilket innebär att den är väl utformad för att kunna användas i framtida arbeten. Vi har även anpassat databasen för multinationalitet, vilket innebär att kunder kan bl.a. välja språk som visas på sidan.

Vi har även lagt in flertalet andra intressanta funktioner i databasen, t.ex. kundrabatter, internationella fraktalternativ och valutor. Detta har vi inte använt i vår nuvarande butik men funktionerna finns i databasen för framtida bruk ifall det skulle behövas. Databasen skulle även kunna anpassas så att annan information skrivs in, t.ex. anställda på företaget.

Administrationsgränssnittet skulle även kunna även utvecklas till Internetaffären, så att enklare administrering kan ske.

5.3 Framtiden för Internetbetalningar

Handeln över Internet har bara existerat sedan 1995, alltså drygt tio år och är därmed inte heller långt kommen. Vi i gruppen tror att Internethandeln kommer fortsätta utvecklas och växa, såväl i Sverige som i resten av världen. Enligt enkäten som genomfördes var det främst äldre människor, som inte växt upp med datorer och teknik på samma sätt som de yngre, som inte handlat över Internet. Därför ser framtiden ljus ut för Internetbutiker. I Sverige där nästan 75% av invånarna har tillgång till Internet hösten 2006, är Internethandeln självklar.

I delar av världen där Internethandel redan är en del av vardagen, Nordamerika, Europa och Asien, kommer förmodligen handeln över Internet handla mer om utveckling än om expandering. Nya typer av butiker och tjänster kommer troligtvis att erbjudas allteftersom tekniken avancerar. Likaså kommer nog butiker att anpassas allt mer till handhållna enheter som handdatorer och mobiler.

Säkerheten är alltid en viktig fråga när det gäller överföringar av pengar och kommer säkert fortsätta att vara så. T.ex. Google och eBay planerar att integrera en funktion på eBay som kallas "Click-to-call". Detta är en funktion som tillåter den som vunnit ett bud på en vara på eBay att trycka på ikonen för "Click-to-call" för att starta ett röstsamtal över Internet via Skype eller Google Talk.

Om fler Internetbutiker följer detta exempel så dröjer det eventuellt inte många år innan man kan sitta hemma i soffan och se och prata med expediten, som man handlar av, med hjälp av videosamtal. Eller att man sitter på bussen med sin mobiltelefon och beställer hem en skjorta från den lokale skräddaren som sedan ligger i postlådan när man kommit hem. Allt vi vet är att Internethandel är en del av nutiden och framtiden.

6 Referenser

6.1 Tryckta källor

Apelkrans, Mats; Åbom, Carita (2001) *OOS/UML*
Studentlitteratur, ISBN 9144021380

Darie, Cristian; Watson, Karli *Beginning ASP.NET 2.0 E-Commerce in C# 2005: From Novice to Professional*
Apres, ISBN 1-59059-468-1

Gulliksen, Jan; Göransson, Bengt (2002) *Användarcentrerad systemdesign*
Studentlitteratur, ISBN 9144020295

Kou, Weidong, Payment technologies for E-commerce
Springer, ISBN 3540440070

Mitrovic, Predrag (2001) *Handbok i IT-Säkerhet*
Pagina Förlags AB, Göteborg ISBN 9163606747

Nielsen, Jacob (1992) *Usability Engineering*
Academic Press, Inc. San Diego, CA, ISBN 0125184069

Norman, D. A. (1988) *The Design of Everyday Things*
London: MIT Press, ISBN 0465067107

Padron-McCarthy, Thomas; Risch, Tore (2005) *Databasteknik*
Studentlitteratur, ISBN 9144044496

Shneiderman, B. (1998) *Designing the user interface*
Addison-Wesley, ISBN 0321269780

Stephens, Ryan K; Plew, Ronald R.; Morgan, Bryan; Perkins, Jeff (1997) *Lär dig SQL på 3 veckor*
Pagina Förlags AB, Göteborg ISBN 9163604973

6.2 Elektroniska dokument

American National Standards Institute
http://www.ansi.org (Acc. 2006-12-07)

Builder
http://builder.com.com/5100-6388-5083541-2.html (Acc. 2006-11-26)

CEKAB
http://www.cekab.se (Acc. 2006-12-07)

DebiTech
http://www.debitech.com/nyhetsarkiv/5.6433d6e310d82ee289d8000880.html (Acc. 2006-11-26)

Department of Computer Science
http://www.cs.umd.edu/%7Eben/Fun-p48-shneiderman.pdf (Acc. 2006-09-22)

Devguru
http://www.devguru.com/technologies/t-sql/home.asp (Acc. 2006-11-26)

Dibs
http://www.dibs.dk/1644.0.html (Acc. 2006-11-26)

ECMA-International
http://www.ecma-international.org/publications/standards/Ecma-262.htm (Acc. 2006-09-22)

Föreningssparbanken
http://www.foreningssparbanken.se/sst/www/inf/out/fil/0,,261568,00.pdf (Acc. 2006-11-17)

Geotrust
http://www.geotrust.com (Acc. 2006-12-07)

Handelsbanken
http://www.handelsbanken.se (Acc. 2006-12-03)

Informit
http://www.informit.com/articles/article.asp?p=31098&rl=1 (Acc. 2006-09-24)

Internet Engineering Task Force
http://www.ietf.org (Acc. 2006-12-07)

International Organisation for Standardisation
http://www.iso.org (Acc. 2006-12-07)

Microsoft
http://www.microsoft.com/sql/default.mspx (Acc. 2006-11-26)
http://msdn.microsoft.com/library/default.asp?url=/library/enus/vbcon/html/vbconadopreviousversionsofado.asp (Acc. 2006-09-24)
http://msdn.microsoft.com/netframework/technologyinfo/overview/default.aspx (Acc. 2006-09-24)
http://msdn.microsoft.com/netframework/gettingstarted/default.aspx (Acc. 2006-09-24)

Moneybookers
https://www.moneybookers.com (Acc. 2006-09-22)

Mssqlcity
http://www.mssqlcity.com/Articles/Compare/sql_server_vs_access.htm (Acc. 2006-11-26)

Netcraft
http://news.netcraft.com/archives/2006/05/17/verisign_to_buy_geotrust_combining_top_ssl_providers.html (Acc. 2006-11-26)

Neteller
http://www.neteller.com/ (Acc. 2006-09-22)

OpenPGP
http://www.openpgp.org (Acc. 2006-12-07)

Paynova
http://www.paynova.com/swe/personal/wallet/functionality.asp?icp=8D42980C67130C3A (Acc. 2006-09-22)

Paypal
http://www.paypal.com (Acc. 2006-09-22)

Payson
https://www.payson.se/prod/default.aspx (Acc. 2006-09-22)

Safeshopping
http://www.safeshopping.org/home.shtml (Acc. 2006-09-22)

Samport
http://www.samport.se/company.asp (Acc. 2006-12-01)
http://www.samport.se/PDF/Samport%20Prislista%20-%20Sverige%20kortversion%20internet%2020060407.pdf (Acc. 2006-12-01)
http://www.samport.se/service_safs.asp (Acc. 2006-12-01)

Techrepublic
http://downloads.techrepublic.com.com/5138-9592-6028761.html (Acc. 2006-11-26)

Usabilitypartners
http://www.usabilitypartners.se/usability/standardssv.shtml (Acc. 2006-09-22)

Useit
http://www.useit.com/papers/heuristic/heuristic_list.html (Acc. 2006-09-22)

Verisign
http://www.verisign.com (Acc. 2006-12-07)
http://www.verisign.se/products-services/security-services/ssl/ssl-information-center/strongest-ssl-encryption/index.html (Acc. 2006-09-22)

Visa
http://partnernetwork.visa.com/pf/3dsec/main.jsp (Acc. 2006-09-22)

W3
http://www.w3.org/Consortium/ (Acc. 2006-11-26)
http://www.w3.org/MarkUp/2004/xhtml-faq (Acc. 2006-09-24)

W3schools
http://www.w3schools.com/sql/sql_join.asp (Acc. 2006-12-03)

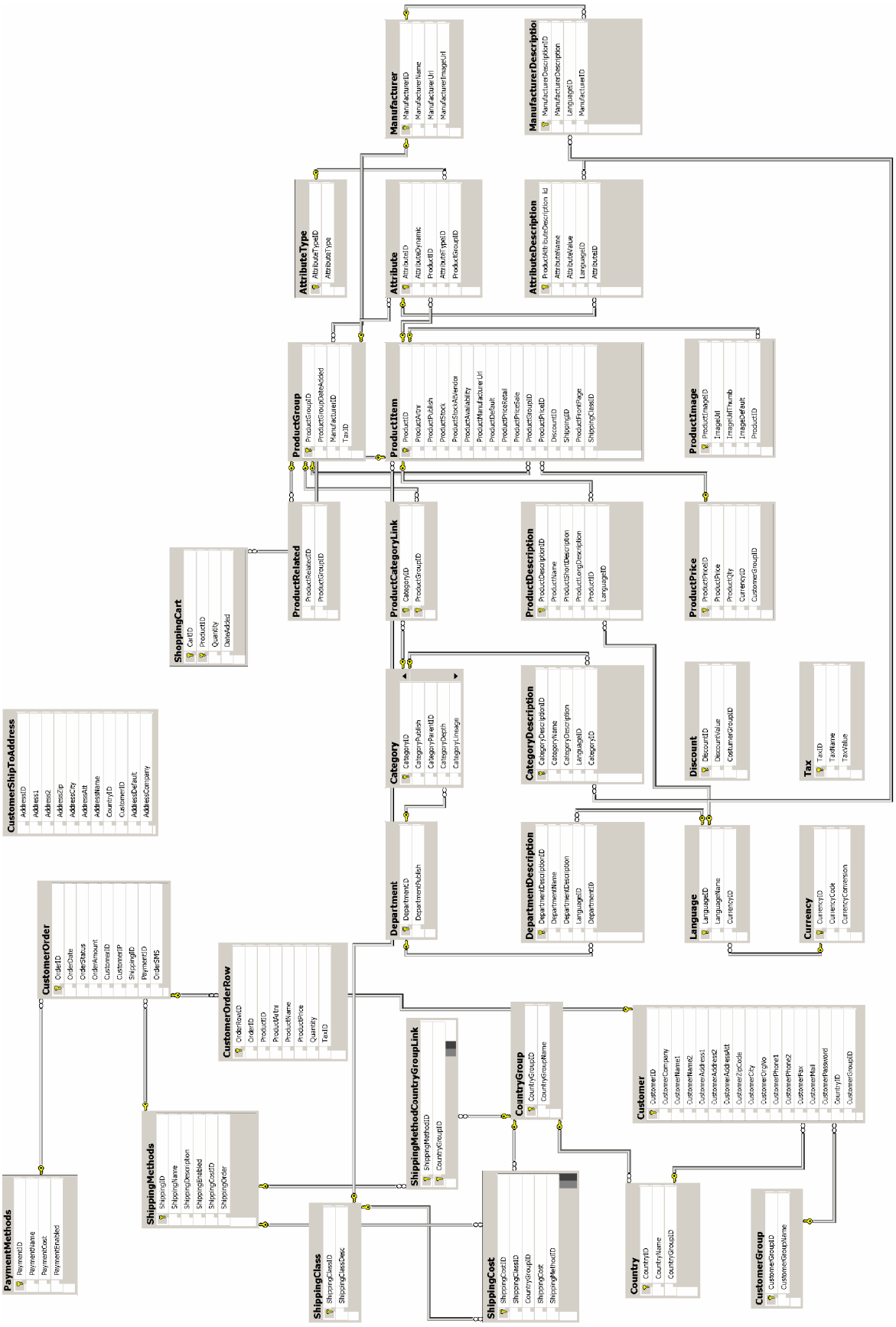
Windowsecurity
http://www.windowsecurity.com/articles/SSL-Acceleration-Offloading-Security-Implications.html (Acc. 2006-12-07)

7 Sökord

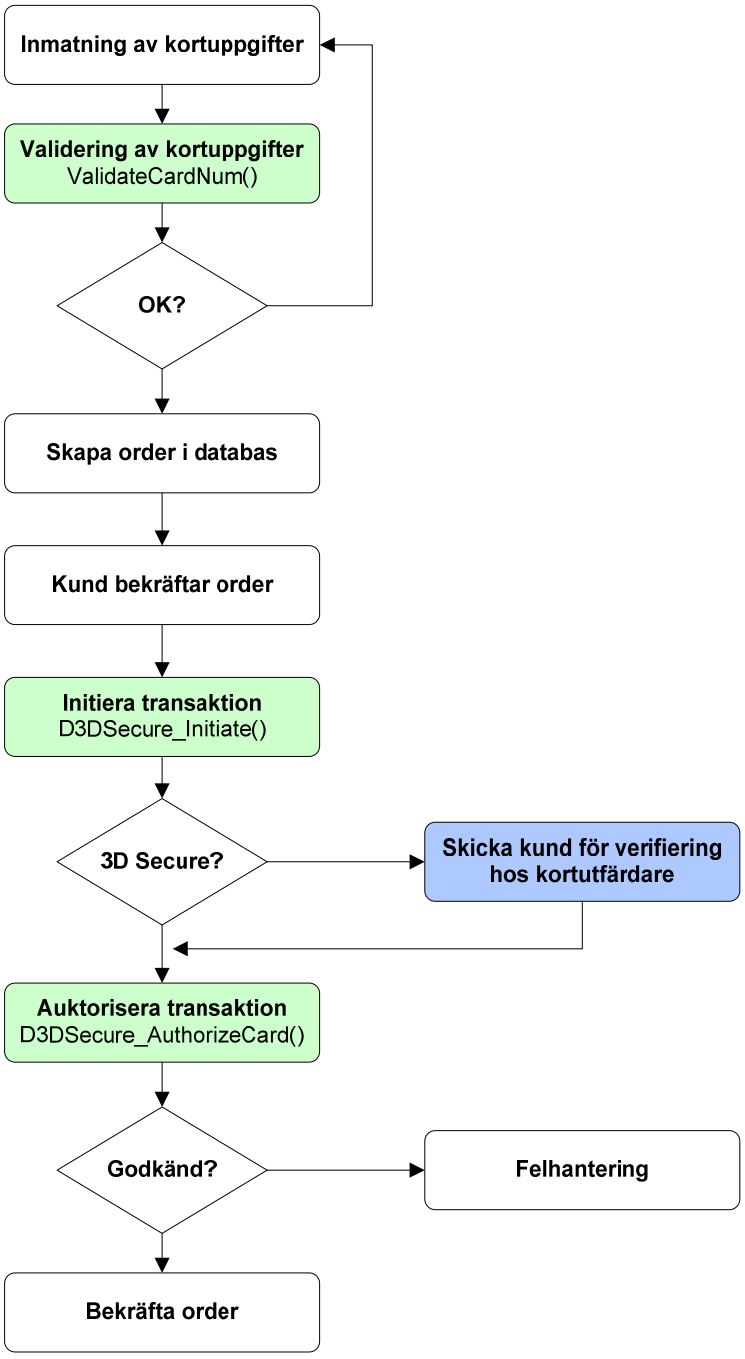
8 Bilagor

Bilaga 1 Databasdesign
Bilaga 2 Kortbetalningsflöde
Bilaga 3 Enkät – Betalning över Internet och säkerhet

Bilaga 1 Databasdesign



Bilaga 2 Kortbetalningsflöde



Bilaga 3 Enkät – Betalning över Internet och säkerhet

Allmänna uppgifter			Ringa in Ditt svar	
Kön:	Man		Kvinna	
Ålder:	18-25	26-39	40-59	60+

1. Har Du någonsin köpt något över Internet?	Ringa in Ditt svar
Ja	Nej
(om ja, gå vidare till fråga 2a, om nej gå vidare till 2b)	

2a. Hur betalade Du köpet över Internet?	Kryssa i rutorna, flera kryss är tillåtet
Via digital plånbok (Paypal, Payson, Paynova, m.fl.)	☐
Via kontokort (VISA, MASTERCARD, m.fl.)	☐
Via Internetbank (Swedbank, SeB, Nordea, m.fl.)	☐
Annat (vad?) ☐	_____

2b. Varför har Du inte köpt något över Internet?	Kryssa i rutorna, flera kryss är tillåtet
Har inte Internet	☐
För svårt att klara av	☐
Litar inte på säkerheten	☐
För dålig service	☐
Vill kunna testa varan i butik	☐
Annat (vad?) ☐	_____

3. Tror Du att Du kommer köpa något över Internet inom det närmaste året?	
Ringa in Ditt svar	
Ja	Nej

4. Hur känner Du för att betala något över Internet? Gör ett kryss

1	2	3	4	5	6
☐	☐	☐	☐	☐	☐
Väldigt osäker					Väldigt säker

5. Vad är viktigast för att Du ska våga betala över Internet?
Värdera med siffrorna 1-4(5 om Du använder eget alternativ)
där 1 är mest viktigt och 4(5) är minst viktigt Varje siffra får bara användas en gång.

Utförlig information om säkerheten bakom köpet ___

Utförlig information om hur köpet går till ___

Välkänd försäljare ___

Rekommendation från bekant ___

Eget (vad?) ___ _____

6. Vad tycker Du webbutiker ska göra för att fler ska våga/vilja köpa något
av dem över Internet?

_____

_____

_____

7. Övriga kommentarer angående betalningar över Internet.

_____

_____

_____

_____

_____

Tack så mycket för hjälpen!