

## Kläddesign på webben

-en prototyp där kunden designar sitt klädesplagg

Victor Wickström

## **Sammanfattning**

Denna rapport är en del av examensarbetet som gjorts i kursen EXB731 - Examensarbete i IT & multimedia 10 poäng. Kursen ingår i programmet Datavetenskap och Medieteknologi vid Växjö universitet. Examensarbetet har utförts på halvtid under vårterminen 2005.

Examinator och handledare är/var Nicholas Pagden.

Beställaren är Helena Tegenfeldt.

Syftet med examensarbetet är att skapa en webbplats där det är möjligt för en kund att designa sitt eget klädesplagg på ett lätt sätt och lägga en order. En designer ska kunna se upp plagget genom att läsa av den information som behövs via en administrativ del av webbplatsen.

I rapporten beskrivs arbetet med webbplatsen. Hur jag har tänkt när jag har skapat den, hur den är uppbyggd och funktionaliteten. Det finns också ett stycke där jag diskuterar slutsatsen av detta arbete. I examensarbetet har jag använt HTML, JavaScript, PHP, SQL, lite CSS och en MySQL databas.

## Innehållförteckning

1. Inledning .....	3
1.1. Syfte .....	3
1.2. Webbplatsens disposition.....	3
1.3. Målgrupp.....	3
2. Beskrivning av tekniker .....	4
2.1. Programmeringsspråket PHP .....	4
2.2. Databasen MySQL.....	4
2.3. JavaScript.....	4
3. Utförandet .....	5
3.1. Prototyper.....	6
3.2. Utvecklingsplattform, språk och programvara .....	7
4. Resultat .....	8
4.1. Utformning och genomgång av webbplats .....	8
4.2. Sidornas funktionalitet .....	16
4.2.1. Bilduppladdning och datumhantering.....	16
4.2.2. Sessionshantering.....	19
4.2.3. Absolut placering .....	19
4.2.4. Funktioner på den offentliga delen av webbplatsen .....	20
4.2.5. Funktionerna på den administrativa delen av webbplatsen .....	24
4.3. Användartester .....	29
5. Nästa sked .....	30
6. Sammanfattning och slutdiskussion.....	31
Referenser .....	33
Bilagor	
Date attribut.	

# 1. Inledning

## 1.1. Syfte

Syftet med projektet är att göra en klädbutik på internet åt en kläddesigner där kunder kan designa sina egna kläder på ett relativt enkelt sätt, utan att ha några kunskaper om hur man designar och syr upp kläder. Webbplatsens administration ska ske helt dynamiskt<sup>1</sup> utan att administratören av denna ska ha någon kännedom om webbprogrammering.

Han ska också kunna skapa, modifiera och ta bort information på webbplatsen som rör utformning av plaggen.

## 1.2. Webbplatsens disposition

Arbetet består egentligen av två delar, en webbplats för kunden och en administrativ webbplats. Webbplatsen för kunden ska vara den delen som är offentlig och där kunden på ett enkelt sätt kan designa sina plagg som han/hon efter vill ha dem.

I fortsättningen av uppsatsen kommer jag istället för att skriva han/hon på alla ställen att bara skriva han.

På den administrativa webbplatsen ska designern kunna lägga upp all information som rör konstruktionen av plagg.

All information<sup>2</sup> mellan den administrativa och offentliga webbplatsen ska lagras i en databas. Bilder som används för uppbyggnad av plaggen ska lagras på webbservern

## 1.3. Målgrupp

Kundmålgruppen är en kvinna eller man mellan 20 och 40 som vill designa sitt eget speciella klädesplagg på ett enkelt sätt via datorn. Personen har ingen större dator- eller klädesdesignvana.

Målgruppen för administrationsdelen är kläddesigner som inte kan programmera, men han kan lite om datorer.

Målgruppen för denna uppsats är min handledare och alla andra som skulle vilja läsa den och ha insikt i mitt examensarbete.

---

<sup>1</sup>Med dynamiskt menas i detta fall att administratören av webbplatsen ska kunna lägga upp nya sorters plagg och delar av ett befintligt plagg så att webbplatsen automatiskt skapar de nya valen i kundens designläge. Till exempel om administratören fyller i att det ska finnas två olika typer av vänsterärmar, då skapas det automatiskt ett alternativ i kundens designläge där kunden kan välja vilken vänsterärm som han vill ha.

<sup>2</sup> All information är i detta fall alla beställningar och alla parametrar till plaggen, dvs. pris, placering av bild, mönster, osv.

## 2. Beskrivning av tekniker

### 2.1. Programmeringsspråket PHP

PHP, som står för Hypertext Preprocessor, är ett skriptspråk för att skapa dynamiska webbplatser i ett HTML dokument. PHP-koden går att blanda med HTML-koden. Webbservern kompilerar först PHP-koden till HTML på servern sedan skickas den färdigkompilerade HTML-sidan till klienten. Detta har gjort att det har blivit lätt att skapa dynamiska sidor och PHP populärt.

Före PHP och andra liknande språk var webbsidorna väldigt statiska, de visade helt enkelt bara upp förutbestämd information på användarens dator. Istället för att kommunicera i en riktning kan man nu kommunicera åt bägge riktningarna och skapa större interaktion med användaren.

PHP skapades av Rasmus Lerdorf 1994, han kallade det för Personal Homepage(PHP). 1994 bytte det namn till Hypertext Preprocessor. (Hall, 2003).

### 2.2. Databasen MySQL

MySQL är ett databasalternativ som lämpar sig att köra med serverspråket PHP. Detta för att både PHP och MySQL körs bäst på en UNIX eller Linux server. En förutsättning för att köra MySQL är att man har en webbservare som klarar av det. Exempelvis Apache eller Microsoft Internet Information Server.

MySQL är utvecklat av ett svenskt företag, MySQL AB, och det har fått bra kritik i olika medier, idag har det vuxit och blivit ett av de största databasalternativen. MySQL licensbestämmelser ligger under GPL, General Public License, med det menas att privatpersoner får använda det fritt om det inte är i kommersiellt<sup>3</sup> bruk/sammanhang.

I MySQL-databasen använder man sig av språket SQL, Structed Query Language, Detta för att lägga till, ta bort och ändra data. (Hall, 2003)

### 2.3. JavaScript

JavaScript(eller EcmaScript som den heter i den standardiserade versionen) är ett skriptspråk som körs i klienten till skillnad från PHP. Till skillnad från HTML som är ett beskrivningsspråk är JavaScript ett programspråk för att lägga till funktioner och göra en HTML-sida mer levande. Det är för att i viss mån göra sidan mer interaktiv och ge feedback till användaren. (Körnefors, 2005)

---

<sup>3</sup> I uppbyggnadsstadiet kommer jag att använda mig av webbhotellet freestarhost.com, om webbplatsen kommer att tas i bruk kommer webbplatsen att flyttas till ett annat webbhotell som har en licensierad MySQL-databas. Men det har jag utelämnat till de som kommer att sköta webbplatsen att ta hand om.

### 3. Utförandet

Jag började arbetet tillsammans med designern Helena Tegenfeldt som kom upp med idén bakom projektet. Hon berättade sin idé och vi diskuterade den för att få samma syn. Den gick ut på att skapa en webbplats på Internet där en kund ska kunna designa sitt eget klädesplagg genom att dra i olika punkter som plagget var uppbyggt genom, som sedan presenterades på en administrativ webbplats för tillverkning.

Målet var alltså att en kund ska kunna gå in på en webbplats och designa sitt plagg och sedan lägga en order av plagget som sedan en designer syr upp.

Eftersom beställaren bara hade en idé och inte visste hur projektet skulle utföras ställde hon inte upp så många kriterier. De kriterierna hon satt upp var:

- att hon ville sköta designen av webbplatsen
- att den ska vara lätt att sköta.
- att designen ska ske med hjälp av att kunden ska dra i olika punkter som bygger upp plagget.

Kundens kriterier är:

- att man själv ska kunna designa de olika plaggen.
- att det ska vara lätt att designa plaggen.
- att det ska vara lätt att lägga en order.

Jag gjorde en efterforskning på Internet om hur webbplatsen skulle kunna möjliggöras.

Detta gjorde jag genom att söka med sökmotorerna <http://www.google.se> och

<http://www.altavista.com> med alla möjliga sökord som hade med projektet att göra. Då

fick jag en bild av hur andra hade löst liknande problemet. Jag frågade också flera personer om hur de skulle kunna tänka sig designa kläder via Internet.

När jag hade fått inspiration och idéer om hur jag skulle kunna skapa webbplatsen började

jag skissa på hur webbplatsens kunddel skulle kunna vara uppbyggd. Idéerna bakom att

designa ett plagg genom rullgardinsmenyer hittade jag på webbplatsen Tailor Online

(<http://www.tailoronline.com/>.) Det som är bra att välja val med hjälp av rullgardinsmenyer

är att du kan välja val i grupperade former men ändå ha en överskådlig bild då valen är

samlade på en relativ liten skärmyta. Det som var mindre bra på den sidan var att den var

svårnavigerad och inte strukturerad. Det gick också bara att designa ett sorts klädesplagg.

En annan inspirationskälla var Jumper.Nu (<http://www.jumper.nu/>). De har valt att

skapa en shockwave-applikation där du kan skapa ditt eget tryck på ett befintligt plagg.

Webbplatsen var stilren och enkel och funktionsduglig, nackdelarna är begränsningarna av

applikationen. Begränsningarna var att det bara gick att skapa ett tryck och detta av text.

Den tredje inspirationskällan som jag har haft är TrendEagle –kepsdesigner

(<http://www.trendeagle.com/flash/index.asp>).

### 3.1. Prototyper

Jag skapade två stycken prototyper som jag presenterade för designern och min handledare.

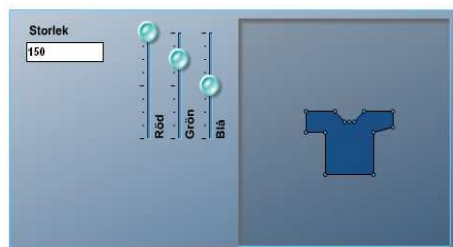


Fig 2.a – Prototyp 1.

Prototyp 1 gjorde jag i Macromedia Flash där kunden skulle dra i punkter för att forma plagget på olika vis. Till vänster av designandet av plagget väljs färg och storlek.

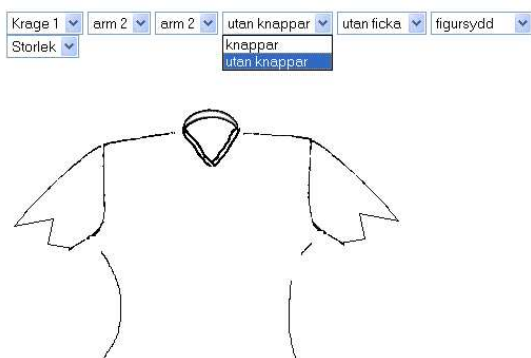


Fig 2.b – Prototyp 2.

Prototyp 2 är skapad i JavaScript där kunden designar sitt plagg genom att välja olika val i rullningslistor.

Jag skapade två olika prototyper för att testa vilket sätt som är bäst att designa ett klädesplagg på. Prototyperna skiljer sig åt på många områden för att jag, uppdragsgivaren och min handledare skulle kunna diskutera för och nackdelar med de olika funktionerna.

Sammanfattningen av det som vi kom fram till i användartestet var följande ting.

Nackdelarna i prototyp 1 är att det är svårt att få något snyggt genom att dra i punkter och om det skulle bli en riktig version skulle det förmodligen komma upp flera felmeddelande för kunden som talar om att det inte går att sy ett plagg på det viset. Fördelen med prototyp 1 är att det ser mer professionellt ut och det kommer alltid att se lika ut i alla webbläsare.

Nackdelen att göra enligt prototyp 2 är att det finns begränsningar i hur kunden kan designa plagget. När man har olika val finns det bara ett visst antal kombinationer. Fördelen att göra enligt prototyp 2 är enkelheten för kunden. Att det är mycket enklare att designa sitt plagg genom att välja olika val.

Slutsatsen blev att vi följer prototyp 2 för att göra det enkelt för användaren. Detta strider mot beställarens kriterier då hennes kriterier var att plaggen ska designas med hjälp av punkter som man ska kunna dra i för att skapa plaggen. Men vi kom överens om att ändra de kriterierna för att göra det lättare för kunden att designa sitt plagg. Om vi skulle ha följt Prototyp 1 skulle kunden ha svårt att designa ett korrekt plagg. Vad skulle då hänt

om kunden fått ett plagg som inte gick att bära för det var felkonstruerat, vems fel skulle det vara i så fall? Kunden såg hur plagget skulle se ut, men han hade inte kunskap om hur plagget skulle vara korrekt designat. Det skulle kunna skapa juridiska och ekonomiska problem.

### 3.2. Utvecklingsplattform, språk och programvara

När jag skapade prototyperna var tanken att jag skulle skapa en shockwave applikation som kördes på en webbserver eller egentligen i klientens webbläsare.

Enligt Macromedia är fördelen med att skapa en shockwave applikation är att det alltid ser lika ut på alla datorer, 97% av alla datorer som är uppkopplade till Internet stöder Macromedia Flashs version av shockwave. (*Macromedia, 2005*)

En annan fördel är att få det snyggt och lättare att få det användaranpassat.

Problemet som dök upp var att programmet Macromedia Flash inte hanterar externa transparenta<sup>4</sup> bilder på ett smidigt sätt. Flash har inget direktstöd av transparenta bilder, om man skulle använda transparenta bilder skulle man få konvertera om dem med olika applikationer för att sedan få lägga in dem som en Flash fil. Alternativet att skapa en shockwave applikation var att använda Macromedia Director men Director har inget direkt stöd för databaser.

Det tredje alternativet var att skapa webbplatsen med hjälp av ett aktivt serverskriptspråk och JavaScript. Problemet är då att absolut placering av alla objekt med JavaScript inte fungerar i alla webbläsare. Absolut placering skiljer sig från den relativa positioneringen genom att objektet som har fått en absolut placering släpper sin statiska placering och låter andra beståndsdelar ta dess utrymme. Objektet med absolut placering skapar ett nytt lager som kan vara över eller under andra objekt, lagret kan man placera med en exakt placering i webbläsarens fönster genom att ange vilken position objektet ska ha i X och Y-led. Detta är viktigt i projektet för att placera bilderna som bygger upp en representation av plagget i designläget för kunden. (*Michaelsson, senast hämtad 2005-05-29*)

Jag valde att använda mig av serverskriptspråket PHP istället för ASP, detta är på grund av att PHP enligt PHPBuddy

- har öppen källkod
- kan köras på flera olika operativsystem till exempel Linux, Solaris och Windows
- är snabbare och mer stabilt.
- är billigare att använda. Exempel om det körs på en kostnadsfri version av Linux och använder en MySQL server(vid icke kommersiellt bruk) behövs det bara betalas för datorn, elen och Internetanslutningen.
- ASP saknar vissa funktioner i standarden, exempel en funktion att ladda upp filer. Men många av de komponenterna kan köpas till i föregående exempel skulle det vara komponenten ASPupload. (*PHPBuddy, 2005*)

En annan anledning till att jag valde PHP är att jag kan få feedback av handledare på skolan då det ges kurser som använder PHP.

Valet jag gjorde var att använda HTML, JavaScript, PHP, MySQL och lite formatering i CSS. Jag använde mig av programvaran Microsoft – Notepad för att skriva all kod, webbläsarna Internet Explorer och Mozilla Firefox för att se resultatet och bildredigeringsprogrammet PhotoShop för att skapa bilder.

---

<sup>4</sup> Transparenta bilder använder jag för att göra det möjligt att bygga upp plagg av bilder i olika skikt. Till exempel vill kanske du ha ett visst mönster och det ska kunna visas igenom de andra bilderna som ligger över den bilden som bygger upp resten (streck som visar armar, kragen, fickor och andra delar) av plagget.

På webbservern använde jag mig av en MySQL-databas. Detta för att skapa en dynamisk sida där det går att lägga ordrar och uppdatera informationen om de olika plaggen. För att kommunicera med MySQL-databasen använde jag mig av SQL-frågor inne i PHP-Sripten. Databasens tabeller ritade jag upp med phpMyAdmin som är en nätbaserad applikation som ligger på webbservern. Tabellerna som jag har skapat består av ett visst antal poster som representeras av rader och attribut som är en kolumn. Jag har valt i de flesta fall att ha en primärnyckel för varje post i varje tabell för att skilja informationen åt.

Jag använde mig av JavaScript för att kontrollera fälten av inmatningsrutorna och uppritandet av plaggen.

Under projektet använde jag mig av webbhotellet 100webspaces <http://freestarthost.com/> som är en gratis webserver som kan hantera PHP och MySQL. Det är meningen att webbplatsen ska med lätthet kunna flyttas till en annan server när webbplatsen kommer till ett kommersiellt ändamål.

## **4. Resultat**

Resultatet består av tre avsnitt. Det första avsnittet handlar om hur webbplatsen ser ut för de som använder den, det andra avsnittet handlar mer om hur sidorna är uppbyggt rent teoretiskt och det tredje avsnittet handlar om användartester.

### **4.1. Utformning och genomgång av webbplats**

Det som jag har skapat är en klädbutik på webben där kunden kan designa sina egna kläder genom att välja olika alternativ i rullningslistor. På den administrativa webbplatsen kan man logga in och sedan skapa, ta bort och ändra plaggen, detaljer och användare.

Jag vill påpeka att den yttre designen av webbplatsen vill Helena Tegenfeldt designa själv, därför har jag inte lagt ner mycket tid på det mer än att designen av webbplatsen ska lätt gå att ändra i framtiden.

Figur 3.a visar den sidan som kunden ser när han har klickat på knappen produkter i den vänstra menyn. Kunden har som val att välja vilken sorts plagg som han vill designa. Dessa plaggen har administratören av webbplatsen skrivit in under administrationssidorna se figur 3.h. Kunden väljer plagget genom att klicka på den bild som tillhör just det plagget.

# Kläddesign

Välj typ av plagg som du vill designa

Linne	hoddie	Skjorta
Linne	Hoddie	Skjorta

**Kund**  
Logga ut  
Ny kund  
Ändra profil  
Mina order

**Handla**  
Produkter  
?? Hjälp ??

**Admin**  
Logga in

fig. 4.a - första valet när kunden ska designa sitt plagg.

När kunden har valt önskvärt plagg kommer han till själva designfasen. Figur 4.b. är en skärm dump som är tagen under denna fasen. I rullningslistorna till höger väljer kunden helt enkelt de olika valen som han föredrar. Dessa rullningslistorna genereras automatiskt fram efter hur administratören har skapat plagget. Detta kommer att beskrivas mer senare i texten.

Kostnaden för plagget visas i en textbox under rullningslistorna. Om kunden vill kan han ladda upp en bild för tryck, detta görs genom att trycka på "Browse" knappen under det plagget som designas.



fig. 4.b – Bild som visar designläget för kunden

# Kläddesign

<b>Kund</b>
Logga ut
Ny kund
Ändra profil
Mina order
<b>Handla</b>
Produkter
?? Hjälp ??
<b>Admin</b>
Logga in



Jag vill beställa  st á 610kr

OBS!! Tryckkostnader är inte inräknade i priset och bilden av trycket är bara en preliminär visning OBS!!

Namn: Victor Wickström

Adress: Stallvägen 46:103  
35256 Växjö

Sverige

Telef: 0735006824

E-post: vwidm03@student.vxu.se

Övrig kommentar

fig. 4.c – visar förhandsgranskningen av det designade plagget före lagd order.

När administratören ska behandla ordena loggar han in som administratör och sedan klickar han Redigera under order i den vänstra menyn. Då kommer det att se ut som figur 4.d. Alltså alla order kommer att listas upp. Genom att klicka på texten "Gå till order" kommer administratören till en liknande sida som figur 4.e.

Användare	Summa	Status	Valj
victor	610.00	Obehandlad Ändra	<a href="#">Gå till order -&gt;</a>
victor	176.00	Mottagen Ändra	<a href="#">Gå till order -&gt;</a>
victor	610.00	Levererad ej betald Ändra	<a href="#">Gå till order -&gt;</a>
hellehansen	176.00	Levererad betald Ändra	<a href="#">Gå till order -&gt;</a>

fig. 4.d – De lagda ordena för administratören av webbplatsen

# Kläddesign

**Användare**  
Redigera  
Ta bort


**Plagg**  
Lägg till  
Redigera  
Ta bort

**Typdel**  
Lägg till  
Redigera  
Ta bort

**Kläddetalj**  
Lägg till  
Redigera  
Ta bort

**Order**  
Redigera  
Ta bort

Logga ut



Del	Beskrivning
Boddy, figursydd	Figursydd, mönster B-1045
Vänsterärm taggig	Vänsterärm taggig sydd - mönster C-1345
Krage V-ringat	krage mönster C-1003
Högerärm taggig	Högerärm taggigt syd - mönster C-4135
Ingen ficka	ingen ficka

Datum för lagd order: 2005-05-30 18:47:08  
Ordern är lagd från IP-adressen: 194.47.122.215  
Kunden har beställt: 1 st  
Summan för produkten är: 610.00  
Kommentar från kunden: Övrig kommentar  
[Bildens tryck](#)

**Kund info för kunden med kundid: 3**  
Victor Wickström  
Stallvägen 46:103  
35256 Växjö  
Sverige  
Telefon: 0735006824  
Email: vwidm03@student.vxu.se

fig. 4.e – visar informationen till administratören av en specifik order.

Om administratören vill skapa en klädesdetalj klickar han på "Lägg till" under Kläddetalj i menyn till vänster. Då får han först välja vilket plagg som han vill lägga till en kläddetalj under. Figur 4.f visar formuläret som kommer upp. På denna sida kan du flytta bilden som representerar plagget genom trycka på knapparna Uppåt, Vänster, Höger, Neråt och/eller skriva in positionen direkt i textrutorna Position i x-led och position i y-led. Representationen av plagget är uppritad samt en bild som inte har någon adress, som i detta fall representeras av ett rött kryss med en rektangel runt (Internet Explorer 6.0). Denna "trasiga" bild är just där bilden/kläddetaljen kommer att placera sig.

**Kläddesign**

Välj plagg ▾

**Användare**  
Redigera  
Ta bort

**Plagg**  
Lägg till  
Redigera  
Ta bort

**Typdel**  
Lägg till  
Redigera  
Ta bort

**Kläddetalj**  
Lägg till  
Redigera  
Ta bort

**Order**  
Redigera  
Ta bort

Logga ut

Uppåt  
Vänster Höger  
Ner

'Klädesplagg': Klädesplagg

Beskrivning: 'Beskrivning av klädesplagg...'

Pris: Pris

Position i X-led: 204

Position i Y-led: 125

Bildfil: Browse...

Typdel: vänsterarm ▾

Lägg till kläddetalj

fig. 4.f – visar hur formuläret ser ut för att skapa en klädesdetalj.

# Kläddesign

<p><b>Användare</b></p> <p>Redigera</p> <p>Ta bort</p> <p><b>Plagg</b></p> <p>Lägg till</p> <p>Redigera</p> <p>Ta bort</p> <p><b>Typdel</b></p> <p>Lägg till</p> <p>Redigera</p> <p>Ta bort</p> <p><b>Kläddetalj</b></p> <p>Lägg till</p> <p>Redigera</p> <p>Ta bort</p> <p><b>Order</b></p> <p>Redigera</p> <p>Ta bort</p> <p>Logga ut</p>	<p><b>Kund-ID:</b> 3</p> <p><b>Förnamn:</b> <input type="text" value="Victor"/></p> <p><b>Efternamn:</b> <input type="text" value="Wickström"/></p> <p><b>Adress:</b> <input type="text" value="Stallvägen 46:103"/></p> <p><b>Postnummer:</b> <input type="text" value="35256"/></p> <p><b>Stad:</b> <input type="text" value="Växjö"/></p> <p><b>Land:</b> <input type="text" value="Sverige"/></p> <p><b>Telefonnummer:</b> <input type="text" value="0735006824"/></p> <p><b>E-postadress:</b> <input type="text" value="vwidm03@student.vxu.se"/></p> <p><b>Användarnamn:</b> <input type="text" value="victor"/></p> <p><b>Lösenord:</b> <input type="text" value="test"/></p> <p><b>Reg. Datum:</b> 2005-03-08</p> <p style="text-align: center;"><input type="button" value="Spara uppgifter"/></p>
---	---

fig. 4.g – visar redigering av kund från administratörens webbplats

# Kläddesign

<p><b>Användare</b></p> <p>Redigera</p> <p>Ta bort</p> <p><b>Plagg</b></p> <p>Lägg till</p> <p>Redigera</p> <p>Ta bort</p>	<p><input type="text" value="Klädplagg"/></p> <p><input type="text"/> <input type="button" value="Browse..."/></p> <p><input type="button" value="Lägg till klädplagg"/></p>
--	--

fig. 4.h – visar skapande av ett nytt klädesplagg, här fylls namnet på klädesplagget i och här kan man också ladda upp en bild som representerar plagget.

## 4.2. Sidornas funktionalitet

På de kommande sidorna kommer jag att beskriva hur funktionaliteten på webbsidorna kommer att fungera, främst kommer jag att ta upp den administrativa delen av webbplatsen och där kunden designar sitt plagg. Jag kommer att ta upp en del kodexempel som är återkommande och viktiga för webbplatsen. Huvudprincipen med den administrativa delen är att uppdatera informationen på kunddelen av webbplatsen, detta genom att fylla i olika formulär och ladda upp bilder till webbplatsen. Detta sker med hjälp av en databas och PHP. När det sker på detta vis behövs det ingen programmeringskunskap för den personen som ska underhålla webbplatsen.

På de olika webbsidorna av den administrativa delen finns det formulär där administratören kan lägga till, ändra och ta bort information. Jag har försökt göra det så enkelt och lättförståeligt som möjligt, därför har jag gjort vissa funktioner visuellt och försökt göra alla sidor liknande av varandra, detta för att vara så konsekvent som möjligt.

Jag har försökt att få ner alla sidor under en skärmssida så att man slipper skrolla.

Jag har försökt följa det som (Buchanan, Marsden & Pazzani 2001 s. 679) nämner att:

- Man bör göra så att det är lätt att navigera med få knapptryckningar,
- Summera informationen med få tecken.
- Använd hierarkiska meny system som användarna känner sig vana vid.
- Undvik att göra så att användaren måste göra en vertikal skrollning.

Detta är skrivit för användandet av apparater med begränsat skärmutrymme. Men jag tycker att de är bra synpunkter i detta fall också förutom kanske just att summera informationen med få tecken för att inte användaren ska misstolka texten.

### 4.2.1. Bilduppladdning och datumhantering

En viktig funktion som används bland annat under skapandet av en ny detalj av ett plagg är uppladdning av bild. När man ska ladda upp en fil till en webbserver med hjälp av PHP behöver man först ett formulär som skickar filen till en PHP-sida. Här nedan är ett exempel på ett sådant formulär.

```
<form method='POST' enctype='multipart/form-data'>
<input type='file' name='minfil'>
<input type='hidden' name='MAX_FILE_SIZE'
value='2000000'>
<input type='submit' value='Skicka'>
</form>
```

*fig. 4.i – HTML för att ett formulär*

Den vanligaste metoden för att skicka formulär är POST. Ett alternativ till det är metoden GET. Det som skiljer dessa två metoder kortfattat åt är att GET skickar med formulärdatan i förfrågningssträngen, det vill säga URL:en. Metoden POST skickar formulärdatan till webbapplikation med hjälp av en dataström via STDIN (standard input). Med andra ord har inte STDIN samma överföringsbegränsning. (Jonsson, 2001:94-95)

Attributen `enctype='multipart/form-data'` under `form`-taggen och attributet `type='file'` under den första `input`-taggen är standard för att skicka filer. Taggen `input` används för formulärfält och den andra `input`-taggen i exemplet ovan innehåller attributet `type='hidden'`, det menas att formulärfältet inte syns för användaren. I detta exempel används den `input`-taggen till att bestämma hur stor bilden som man skickar upp får vara. Filstorleken skrivs in i `value`-attributet, om inget skrivs in här blir det automatiskt maximal överföring av fil på 2 MB. Jag har angett 2000000 vilket motsvarar 2000000 byte som är

ca 2MB, anledningen att jag har skrivit med det är det ska bli lättare att gå in i och läsa av och ändra koden i framtiden. Formuläret skickas till ett PHP-dokument. Figuren nedan visar koden som jag använder för att spara filen på webbservern.

```

if ( isset( $_FILES[ "minfil" ] ) )
{
    # en array med de filtyper som vi kan godkänna
    $godkända_typer[] = "image/gif" ;    # gif
    $godkända_typer[] = "image/jpeg" ;  # jpeg
    $godkända_typer[] = "image/jpg" ;   # jpeg
    $godkända_typer[] = "image/x-png" ;  # png

    # visar infot från $_FILES
    echo "\$_FILES[ \"minfil\" ][ \"name\" ] " .
        $_FILES[ "minfil" ][ "name" ] . "<br>";

    echo "\$_FILES[ \"minfil\" ][ \"type\" ] " .
        $_FILES[ "minfil" ][ "type" ] . "<br>";

    echo "\$_FILES[ \"minfil\" ][ \"size\" ] " .
        $_FILES[ "minfil" ][ "size" ] . "<br>";

    echo "\$_FILES[ \"minfil\" ][ \"tmp_name\" ] " .
        $_FILES[ "minfil" ][ "tmp_name" ] . "<br>";

    echo "\$_FILES[ \"minfil\" ][ \"error\" ] " .
        $_FILES[ "minfil" ][ "error" ] . "<br>";

    # om det inte är ett fel och om filtypen är någon av de godkända
    # så flyttar vi den till samma katalog som vi finns i
    if ( $_FILES[ "minfil" ][ "error" ] == 0 AND
        in_array( $_FILES[ "minfil" ][ "type" ], $godkända_typer ) )
    {
        # kontrollera om den går att flytta från sin temporära plats
        # idetta exempel laddar jag upp bilden till en katalog
        # med namnet mina_bilder
        if ( move_uploaded_file($_FILES[ "minfil" ][ "tmp_name" ],
            "../..user/produkt/" . $_FILES[ "minfil" ][ "name" ] ) )
        {
            # visa om det gick
            echo "Filen " .
                $_FILES[ "minfil" ][ "name" ] .
                " uppladdad <br >";

            // Nästa rad om man vill ta bort bilden
            // unlink ( "../..user/produkt/" . $_FILES[ "minfil" ][ "name" ] );
        }
        else
        {
            # nåt skumt håller på att ske?
            echo "Bilden misslyckades att laddas upp";
        }
    }
}

```

fig. 4.j – visar koden för uppladdning av bilder

Den första if-satsen kontrollerar om fältet "minfil" innehåller en fil. Alltså om användaren har fyllt i en bild i formuläret som skickades. Om villkoret stämmer läggs godkända format på bilder in i en array som heter godkända\_typer, detta är för att sedan kontrollera att det bara är bilder som laddas upp. De kommande echo raderna skriver ut vilket namn, vilken typ, vilken stolek i bytes, det temporära namnet och felmeddelandet som bilden har, men detta är bortkommenterad i den slutgiltiga version av webbplatsen.

Nästa if-sats if "( \$\_FILES[ "minfil" ][ "error" ] == 0 AND in\_array( \$\_FILES[ "minfil" ][ "type" ], \$godkända\_typer ) )" kontrollerar om antal inträffade fel är 0 och filtypen finns med i arrayn som vi skapade nyss. Om det villkoret stämmer kontrollerar den om det går att flytta filen till katalogen ../..user/katalog/.

Unlink kommandot kan man använda för att ta bort filen som precis är uppladdad.

(Jonsson Viktor, 2001, s.213-214)

Om det inte gick flytta filen från det temporära biblioteket till den aktuella katalogen skrivs felmeddelandet "Bildens laddning misslyckades att laddas upp" ut på skärmen.

Kommandot Unlink används i fler tillfällen i koden på webbplatsen, bland annat använde jag den för att ta bort bilder som kunder har laddat upp som inte ingår i en order. Denna koden ligger i webbsidan högst upp på webbplatsen i den webbsidan där webbplatsenslogga befinner sig i. Anledningen att jag la den där är att det är en sida som alla som besöker webbplatsen kör och det oftast bara en gång.

```
$result = queryDB("SELECT * FROM bild_temp",
$link);
while ($rad = mysql_fetch_assoc($result))
{
    $tmp_datum = intval($rad['datum']);
    $datum     = intval(date('Ymd')) - 1;

    /// Kollar om det är en bild som är mer än en
dag
    /// gammal i tmpdatabasen
    if ($tmp_datum < $datum)
    {
        $bildstig = split("../../", $rad['stig']);
        $stig = $rad['stig'];
        unlink ( $bildstig[1] );
        $query = "DELETE FROM bild_temp WHERE
stig='$stig' LIMIT 1";
        $result2 = queryDB($query, $link);
    }
}
```

*fig. 4.k – PHP kod för att ta bort gamla bilder*

Det som koden här ovan gör är att den öppnar databasen och går igenom alla bildstigar<sup>5</sup> som ligger under tabellen bild\_temp. När den stöter på en fil kontrollerar skriptet om datumet på filen är mer än en dag gammal (egentligen 2 dagar gammalt). Om villkoret stämmer tas bilden bort från webbservern och databasen. Det som inte är nämnt här är att när en bild laddas upp i designläget av plagget under kundens webbplats så läggs datumet in först i filnamnet och sedan in i databasen. Se kommande kod

```
$datum = date("YmdHis");
```

*fig. 4.l – PHP-kod för att få fram datum*

Koden här över skapar en variabel av typen sträng med ett datum i på följande sätt ÅÅÅÅMMDDTTMMSS exempel 2005050524011232. Detta är för att få ett unikt datum. Men för att göra det ännu mer unikt läggs det till filnamnet efter, exempel 2005050524011232bild.jpg. När det är klart kopieras den som föregående exempel där man laddade upp bilder, med undantaget koden nedan då datumet är inlagt.

```
move_uploaded_file($_FILES[ "minfil" ][ "tmp_name" ],
```

---

<sup>5</sup> bildstigar – minnesadresser till bilder

```

    ../../user/produkt/bilder/" . $datum . $_FILES[ "minfil"
  ][ "name" ])

```

*fig. 4.m – PHP-kod för flytta en bild*

Se bilagor för alla attribut i funktionen date());

#### 4.2.2. Sessionshantering

På nästintill alla sidor finns det någon form av sessionshantering. Sessionshantering möjliggör bevaring av variabler mellan de olika webbsidorna på webbplatser. Tack vare det kan vi komma åt information var vi vill på ett enkelt sätt. Alternativet till Sessioner är Cookies. Cookies fungerar genom att en liten textsträng sparas på klientens dator (oftast den datorn där webbläsaren körs ifrån). En av anledningarna att jag valde Sessioner istället för Cookies är att Sessioner fungerar för alla, detta är för att sessionerna körs och lagras på servern. En annan är att enligt svensk lag måste man informera användaren att webbplatsen innehåller cookies och då kanske någon kund blir skrämmd och går vidare till en annan webbplats på Internet. Den tredje anledningen är att det är många som ogillar cookies och därmed har stängt av den funktionen i sina webbläsare.

För att använda cookies på en webbsidan måste man börja skriva koden `session_start()`; högst upp i dokumentet.

För att sedan skapa en variabel använde jag mig av kommandot `session_register("kund_log");` kund\_log är i detta fallet variabelnamnet. För att tilldela sessionsvariabeln gör jag liknande som att tilldela en vanlig variabel. Exempel `$_SESSION["kund_log"] = 'ja';`

Nu har jag tilldelat kund\_log sessionen en sträng som är 'ja'. På min webbplats betyder det att kunden är inloggad.

För att döda sessionerna använder jag mig av kommandot `session_destroy()`; Då försvinner alla variabler och detta gör jag bland annat när en användare loggar ut.

Jag använder sessioner till att lagra om kunden och administratören är inloggad eller inte. Jag använder det också till att hålla reda på vilken kund som är inloggad.

#### 4.2.3. Absolut placering

Jag använder absolut placering av bilden med hjälp av JavaScript bland annat i kundens designläge av plaggen. Koden här under placeras inom head-taggar i html-koden.

```

<script type="text/javascript">
<!--
function bytbild(bild,namn)
{
bild2 = bild.split("#")
document.getElementById(namn).style.position="absolute"
document.getElementById(namn).style.left=bild2[1]
document.getElementById(namn).style.top=bild2[2]
document.getElementById(namn).src = bild2[0]
}
-->
</script>

```

*fig. 4.n – Kod för absolut placering i Javascript*

bytbild är en funktion som har två inparametrar, bild och namn. Namn är bild-taggens idnamn, bild är en sträng som innehåller källadressen till bilden, vänsterpositionen<sup>6</sup> och

<sup>6</sup> vänsterpositionen – position räknat från vänsterkanten av webbsidan , i detta fall i pixlar.

toppositionen<sup>7</sup>. Strängen använder ett #-tecken som avskiljer, funktionen split delar upp strängen i en array som heter bild2.

```
<SELECT id="0a" NAME="0"
onChange="byt bild(this.options[this.selectedIndex].value
, '0');">
<script type="text/javascript">
<!--
    document.getElementById("0a").style.position="absol
ute"
    document.getElementById("0a").style.left="400"
    document.getElementById("0a").style.top="35"
-->
</script>

<option value=" ../produkt/test4.gif#0#20#21#100">Boddy,
figursydd<br>
<option value=" ../produkt/test2.gif#0#20#22#100">Boddy,
vanlig<br>
</SELECT>
```

fig. 4.o – Kod för absolut placering i Javascript som är genererad med PHP

PHP genererar följande kod i designläget där kunden designar plagg. Det som koden gör är att den skapar en rullningslist som har två alternativ (Boddy figurssydd och Boddy vanlig), de två alternativen genereras från en databas som administratören har skickat in information i. Rullningslistens placering anges av JavaScript-funktionen enligt figur 4.o. Position är absolut, med det menas att den får en exakt position på webbsidan angivet i position från vänster och från toppen av sidan. Just denna absolut placeringsmetod fungerar inte i alla webbläsare med alla objekt. Dessa Select satserna (rullningslisterna) är just ett sådant objekt som inte fungerar i till exempel Mozilla Firefox.

När ett val är gjort av användaren i rullningslistan anropas funktionen bytbild, som beskrivs föregående i texten, Värdet som skickas in är det som står efter *<option value=* som tidigare nämns är #-tecknet avskiljare. Om exempel användaren väljer *Body, figursydd* skickas strängen  *../produkt/test4.gif#0#20#21#100* och en 0 in i funktionen bytbild. Nollan är rullningslistans Namn. Nu aktiveras bytbild-funktionen och bilden samt positionen av bilden ändras.

#### 4.2.4. Funktioner på den offentliga delen av webbplatsen

##### Ingångssidan ramsidan(index.htm)

Denna webbsida är enkel och är skriven totalt i html. Webbsidan innehåller tre ramar, en ram överst där logotypen är, en ram till vänster som är länkad till menyn och en höger ram som är länkad till en ingångssida som designern Helena Tegfeldt själv vill designa och skriva.

Jag valde att lägga upp webbplatsen med hjälp av ramar på detta sätt för att det ska var lätt att ändra om designen efteråt. Layouten av att bygga upp webbplatsen med tre ramar används ofta ute på Internet och är en vanlig uppsättning. Det och att det föll naturligt in var anledningen till att jag använde den Layouten. Anledningen till att jag la logotypen överst var att ögat ska se den direkt och kunden ska se direkt vart han har kommit. Menyn

---

<sup>7</sup> toppositionen – position räknat från toppen av webbsidan, i detta fall i pixlar.

placerade jag till vänster, detta för att många användare är vana att ha menyn till vänster och det kommer då kännas naturligt att ha menyn till vänster.

### **Menyn (meny.php)**

Detta är webbplatsens vänstra ram. När man loggar in anropas ett Javascript som laddar om denna sida. I menysidan finns det sedan ett PHP-skript som kontrollerar om användaren är inloggad, om användaren är inloggad visas knappen "Logga ut" annars visas knappen "Logga in". De olika länkalternativ som finns är uppdelade i 3 kategorier, den första kategorin är Kund och länkalternativen är:

- Logga in eller Logga ut
- Ny kund
- Ändra profil
- Mina order

Under kategorin Handla finns två länkar:

- Produkter
- ?? Hjälp ??

Under den sista kategorin Admin finns bara en länk.

- Logga in

Jag har valt att bara ha 7 länkar i menyn och det är för att enligt inUse, 2004, klarar bara korttidsminnet av att klara av komma ihåg mellan 5-9 saker samtidigt, därför ska man försöka skala bort så mycket oväsentlig information som möjligt. För att underlätta minnet kan man genom att skriva ord som vi känner igen. Ett annat sätt att underlätta för minnet är att just gruppera precis som jag har gjort här över.

Detta kanske känns kanske lite meningslöst när man ser menyn hela tiden, men det är lättare att hitta det man vill och det går snabbare att navigera. Därför har jag försökt att skala ner, gruppera och använda ord som de flesta känner igen. Jag har valt att skriva hjälp med flera frågetecken för att om du som användare vill ha hjälp på sidan ska du inte behöva leta efter en text utan att det ska stå ut lite.

Knapparna är utformade på ett vis så att när man håller musmarkören över en knapp i menyn byts bilden ut. På det sättet talar det om för användaren att någonting händer här. Det blir också en handikon med pekfingret uppe när pekdonet kommer över en klickbar bild, detta för att tala om för användaren att det är en länk.

### **Inloggning (login.php)**

För att lägga en order måste kunden vara inloggad. Detta gör kunden genom att klicka på "logga in" i menyn. Inloggningen är uppbyggd av två textfält, det första textfältet är till för att skriva in användarnamn och det andra textfältet är till för att skriva in lösenord. Om allt är ifyllt klickar användaren på knappen "Logga in", då laddas webbsidan om med inparametrarna som är ifyllda i textfälten. Webbsidan är kopplad till en databas och jämför om användaren har skrivit in korrekta uppgifter. Ifall uppgifterna är korrekta blir kunden inloggad och en session skapas som håller reda på det, om uppgifterna är felaktiga kommer det upp ett felmeddelande. Om inte kunden är registrerad kan han trycka på knappen "Registrera dig", då laddas en sida som kunden kan registreras på.

Jag har valt att lägga upp det på detta sätt på grund av tre anledningar, den första anledningen är för att kunden bara ska behöva fylla i sina uppgifter en gång, den andra anledningen är att få in en liten säkerhetsåtergård. Den tredje anledningen är att användarna ska på ett enklare sätt kunna spåra sina beställningar.

### **Registrering av ny kund (kundupg.php)**

För att en kund ska kunna beställa ett plagg måste han registrera sig. Detta gör han genom att klicka på "Ny kund" i menyn. Registreringssidan består av några fält, dessa fält är:

förnamn, efternamn, adress, postnummer, postort, land, telefonnummer, e-postadress, användarnamn, lösenord samt bekräftelse av lösenord. Jag har valt att lägga textrutorna i två spalter för att få det naturligt. Det vill säga att till höger om förnamn finns efternamn, till höger om postnummer finns postort osv. Rutorna är lagda som om man skulle skriva adressen på ett brev och posta det i en brevlåda. När kunden har fyllt i uppgifterna kontrolleras fälten med hjälp av PHP, anledningen att jag har valt att kontrollera uppgifterna med PHP istället för JavaScript är att JavaScript kan stängas av i webbläsaren och då komma igenom kontrollen.

Jag använde mig av den inbyggda funktionen `ereg()`, alltså reguljära uttryck. Funktionen `ereg()` fungerar på följande vis, `ereg(mönster, text_som_ska_valideras, resultat_lista)`. Där mönster är hur texten ska se ut, `text_som_ska_valideras` är den texten som ska sökas och `resultat_lista` är en array där de uppdelade resultaten sparas i. Ett exempel på följande beskrivning kan se ut så här: `if (ereg("^(.+)\.(.+)$", $_POST['epost'], $matchvektor))`. Det ifsatsen gör är att den kontrollerar så att det först kommer tecken sen ett @ med följt av fler tecken och en punkt och ännu mer tecken, texten som söks igenom är skickat till sidan via POST och har attributnamnet 'epost'. (Jonsson, 2001:164-178)

Om alla uppgifter är ifyllda på ett korrekt sätt sparas uppgifterna samt IP-adressen i databasen så att kunden kan logga in i framtiden. Det kontrolleras också att det inte finns någon annan användare med samma användarnamn i databasen. Ifall uppgifterna är ifyllda på ett felaktigt sätt får användaren ett felmeddelande.

Anledningen till att jag valde att skapa en verifiering av de olika textfälten är för att minimera falska användare. Det går att komma förbi detta genom att fylla i falska uppgifter och det är därför som jag har valt att spara IP-numret ifall användaren skulle behövas spåra i framtiden, vid eventuellt falska ordrar. Man skulle kunna öka säkerheten extra men jag anser att det inte behövs vid just denna webbplats.

### Ändra kunduppgifter (andrakundupg.php)

För att kunna ändra kunduppgifterna måste kunden vara inloggad, om inte kunden är inloggad kommer kunden automatiskt till inloggningssidan, det genom följande kod.

```
<SCRIPT LANGUAGE="JavaScript">
<!--
    alert("Du måste logga in först");
    parent.frames[2].location.href="login.php";
-->
</SCRIPT>
```

*fig. 4.p – JavaScript för att ladda en webbsida i en annan ramsida.*

Koden över är JavaScript och det den gör är att en dialogruta öppnas med texten "Du måste logga in först". Ram 2 som är definierad i `index.htm` laddar sidan `login.php`. Med andra ord är det samma ram som Ändra kunduppgifter körs i och det är inloggningssidan som laddas.

Jag har valt att lägga in detta för att göra det lättare för kunden, att han ska slippa leta sig fram till inloggningssidan.

När kunden har loggat in och vill ändra sina uppgifter trycker han på knappen "Ändra profil" i menyn. Webbsidan som visas är liknande som "Ny kund"-sidan. Det som skiljer sig åt är att alla textrutorna redan är ifyllda efter vad som står i databasen. Det som inte fylls är lösenordsrutorna. Det finns heller ingen ruta för att ändra användarnamnet men det står utskrivet på skärmen i ren text. En återställnings-knapp finns som återställer rutorna efter den existerande databasen. Kontrollen skiljer sig också lite från "Ny kund"-sidan genom

att när kunduppgifterna ändras kontrolleras inmatningsrutorna också av JavaScript för att få en snabb feedback till kunden. Anledningen av att jag tog bort dubbelkontrollen från ”Ny kund”-sidan var att det upplevdes störande av vissa vid användartestet då det kom upp dialogrutor hela tiden. Detta upplevs inte på samma vis under ändrandet av kunduppgifterna då de flesta rutorna redan var korrekt ifyllda. Något som upplevdes positivt under användartestet av dessa webbsidor var att felmeddelandena var informativa och användartestarna förstod vad det var som var fel inskrivit.

### **Mina order (order.php)**

För att en kund ska kunna undersöka hans lagda order måste han vara inloggad, om han inte är inloggad kommer inloggningssidan upp automatiskt. Inloggningsrutan kommer upp för att göra det lättare för användaren att navigera.

Webbsidan som presenterar kundens lagda ordrar är kopplad till databasen och informationen om summan, status på order och vilket datum ordern är lagd hämtas från varje order där just den kunden som är inloggad har lagt. Det finns också en knapp med etiketten ”Visa” till höger om varje order. Jag har valt att lista upp de olika orderarna på det sättet för att få en mer överskådlig blick på alla ordrar som kunden har lagt. För att visa mer specifikt om just en order klickar man på den knappen och sidan orderi.php öppnas. Överst på sidan presenteras en bild på det designade plagget samt övrig information om beställningen.

Anledningen till att jag har gjort denna funktion att kunden ska kunna kolla på sina lagda ordrar är för att kunden ska kunna se att en order lagd, status på ordern samt om han har lagt en dubbelorder.

### **Produkter (produkter.php)**

Jag har redan tagit upp de flesta funktionerna på denna webbsidan, därför kommer jag att bara beskriva denna sidan kortfattat.

Genom att klicka på ”Produkter” i meny kommer man till en sida där de olika typerna av plagg presenteras. Genom att klicka på tillhörande bild av det plagg som man vill designa öppnas designläget för det plagget.

Designläget är uppbyggt av att PHP genererar rullningslistor av de olika delarna av plaggen som bygger upp plagget. Rullningslistornas placering är till höger av bilderna som representerar plagget, denna placering genereras fram av PHP från databasen och placeras med hjälp av absolutplacering med hjälp av JavaScript. Till vänster av rullningslistorna finns det bilder som representerar olika delar av plaggen. Dessa delar bygger upp ett komplett plagg och de placeras som rullningslistorna med hjälp av absolut placering i JavaScript. Genom att ändra val i rullningslistan ändras bilden av den delen av plagget som representerar just valet i rullningslistan. Detta sker med hjälp av JavaScript. Det går också att skapa ett tryck och detta gör man genom att ladda upp en bild. Under avsnittet Bilduppladdning och datumhantering finns en större förklaring med bilder.

Efter att ha designat klart plagget laddas sidan om och en förhandsgranskning visas. Här ser man all information som kunden behöver veta om beställningen, det går också att skriva hur många exemplar av plagget man vill beställa samt övriga kommentarer. När plagget beställs läggs ordern in i databasen.

Jag har valt att lägga upp produkt och beställningen på flera steg för att få en lättare och mer strukturerad procedur. Jag har valt att de olika typerna av plaggen ska presenteras av både en bild och text för att det ska vara lättare för kunden att veta vad det är för plagg som är möjliga att designa.

Slutsatsen av val av prototyperna var att själva designsidan skulle vara uppbyggd av rullningslistor med olika alternativ som ändra en bild och alla bilder representerade det fulla plagget. Därför har jag valt att bygga upp designsidan på just det sättet. I prototypen

hade jag placerat alla rullningslistor över bilderna som representerar plagget men i den färdiga produkten har jag placerat de till höger om bilden och det är för att få plats mer information och val på en skärmsida så kunden slipper att "scrolla" i vertikal och horisontell riktning.

Jag har valt att lägga till ett steg som för förhandsgranskning av ordern innan beställningen är skedd för att kunden ska kunna gå igenom alla uppgifter och se en förhandsgranskning av trycket innan beställaren beställer plagget.

### **Hjälp (hjalp.php)**

Hjälp-sidan är en statisk webbsida som innehåller information som hjälper kunden genom alla processer. Jag har valt att göra denna sida för att hjälpa användaren när den har kört fast eller vill ha hjälp.

### **Logga ut (loggut.php)**

Detta har jag gjort på grund av två anledningar, säkerhet och underlätta för servern som webbplatsen kör på. Med säkerhet menar jag att ingen annan vid samma dator ska kunna beställa något åt den kunden som är inloggad efter att kunden har lämnat datorn. Den andra aspekten är att underlätta för servern som slipper att ha kvar alla sessioner och ta upp minnesplats.

Vid utloggning raderas alla sessioner och kunden är utloggad ur systemet.

## **4.2.5. Funktionerna på den administrativa delen av webbplatsen**

För att logga in på den administrativa delen av webbplatsen klickar du på "Logga in" under "Admin" i menyn.

### **Inloggning (log\_in.php)**

Administratören har fått ett eget användarnamn och lösenord, som i sin tur är inbakad i koden för att få det extra säkert. För att logga in skriver administratören in sitt användarnamn och lösenord i de två textfälten som finns under inloggningssidan. Ifall administratören skriver in fel användarnamn eller lösenord kommer ett felmeddelande upp, för att tala om att lösenordet är fel. Om användarnamnet och lösenordet är rätt skapas det en session på servern som talar om att administratören är inloggad och menyn laddas om med JavaScript.

### **Redigera användare (uppdatera.php)**

Under denna webbsida listas alla registrerade kunder. För att ändra informationen för en specifik användare klickar du på texten "Ändra" vid den önskade användaren. Webbsidan laddas om och hämtar informationen för just den användaren från databasen. Här kan administratören redigera all information som han vill om användaren.

Jag har valt att göra denna funktionen för att administratören/designern ska kunna kolla upp alla användare samt att redigera informationen om användaren.

### **Ta bort användare (tabort.php)**

Utseendet för denna webbsida är samma som den sidan där man redigerar användaren, den enda skillnaden är att det står "ta bort" istället för "Ändra". Vid klickning av "Ta bort" vid en specifik registrerad kund raderas kunden från databasen.

### **Lägg till ett plagg (lägg\_till.php)**

För att lägga till ett klädesplagg som kunden ska kunna designa fyller administratören i vad klädesplagget ska heta och ladda upp en bild som kunden sen kan klicka på för att designa just det plagget.

Bilden laddas upp med hjälp av PHP och bilden placeras på webbservern. Koden som används är liknande som den jag har beskrivit tidigare i rapporten.

### **Redigera ett befintligt plagg (uppdatera.php)**

Alla webbsidor där man redigerar något är upplagt på samma vis för att inte rör till det för administratören. På den första sidan listas alla klädesplagg, för att ändra om ett plagg väljer man just det plagget och fyller i de uppgifterna som ska ändras och databasen uppdateras.

### **Ta bort ett befintligt plagg (tabort.php)**

Alla ”ta bort” sidor fungerar på ett liknande sätt. När man tar bort ett klädesplagg raderas all information om plagget och dess beståndsdelar (typdelar och klädesdetaljer). Detta görs för att inte databasen och webbservern ska fyllas med massor av orelaterade objekt.

### **Lägg till typdel (lagg\_till.php)**

Jag har valt att bygga upp designfunktionen av klädesplaggen genom att skapa bilder i olika skikt/lager. Detta är för att det ska gå lättare för administratören att skapa plagg. I det understa lagret rekommenderar jag att administratören ska lägga en bild av olika mönster. Detta för att sedan lägga flera lager på varandra av transparenta bilder som ska representera olika delar av plagget. Om det är gjort på ett korrekt sätt kan kunden välja om han vill exempel ha ett blått mönster av flis. Det understa lagret har värdet 0 och sedan går det uppåt. Alltså en typdel är ett skikt och för att lättare veta vad det är för skikt namnges skiktet här under. Exempel istället för att hålla reda på att högerärmen ligger i skikt 5 kan man ha skapat en typdel som heter högerärm och sedan när administratören skapar en klädesdetalj välja högerärm som skikt/lager.

Det finns en annan anledning till att skapa olika skikt och det är att rätt klädesdetalj ska komma i rätt rullningslist för kunden i kundens designläge.

Denna webbsida för att skapa en typdel är uppbyggd av ett formulär av tre rutor. Den första textrutorna ska man fylla i vad typdelen ska heta, exempel högerärm. I den andra textrutorna ska man fylla i en siffra som representerar vilket lager typen ska vara i. Det understa lagret är 0, sedan 1 osv. Den tredje rutan är en rullgardinsmeny där man ska välja vilket klädesplagg som typdelen hör till.

Webbsidan fungerar genom att ett formulär skapas, under rullgardinsmenyn hämtas data från databasen. När alla uppgifter är ifyllda skapas det en ny post i databasen.

Jag har valt att strukturera det på detta sätt för att få struktur och att det ska lätt gå att ändra utan att administratören/designern ska behöva kunna någon HTML.

### **Ändra typdel (uppdatera.php)**

Alla typdelar listas på webbsidan för att man sedan ska kunna välja vilken typdel man vill redigera. När man väl har valt en typdel kommer det upp ett formulär där uppgifterna kan ändras. Webbsidan hämtar data från databasen och uppdaterar dem.

### **Ta bort typdel (ta\_bort.php)**

Denna webbsida fungerar på ett liknande sätt som de andra ta bort sidorna. När en typdel tas bort raderas alla klädesdetaljer som ligger under denna typdelen. De funktionerna som utförs på denna webbsidan är att data hämtas från databasen och vissa utvalda poster raderas.

### Lägg till klädesdetalj (lägg\_till.php)

När administratören lägger till en klädesdetalj måste han först välja under vilket plagg som klädesdetaljen ska skapas under. Detta val görs i en rullgardinsmeny.

När väl detta är gjort är det dags att fylla i ett formulär om informationen av plagget. Det som behövs fyllas i är: Namn på detalj, beskrivning, pris, position i x-led, position i y-led, bild och typdel.

Beskrivning av bilden är tänkt att bara administratören ska kunna se och är till att för att beskriva vilket mönster som ska användas och annan viktig information som behövs för att tillverka plagget.

Positionen sker i absolut placering och kan antingen fyllas i textrutor eller kan man flytta en bild visuellt på ett redan uppritat klädesplagg. Förflyttningen sker med hjälp av JavaScript och funktionerna som styr detta ser ut på följande vis.

```
<script type="text/javascript">
<!--
  POSTOP = 0;
  POSLEFT = 0;

function SETPOSY(top)
{
  POSTOP = top;
}

function SETPOSX(left)
{
  POSLEFT = left;
}

function Upp(namn)
{
  POSTOP = document.getElementById("POSY").value;
  var top = 1;
  top = POSTOP - top;
  SETPOSY(top);
  document.getElementById(namn).style.position="absolute";
  document.getElementById(namn).style.top= POSTOP;
  document.getElementById("POSY").value= POSTOP;
}

function Ner(namn)
{
  POSTOP = document.getElementById("POSY").value;
  var top = -1;
  top = POSTOP - top;
  SETPOSY(top);
  document.getElementById(namn).style.position="absolute";
  document.getElementById(namn).style.top= POSTOP;
  document.getElementById("POSY").value= POSTOP;
}

function Vanster(namn)
```

```

{
POSLEFT = document.getElementById("POSX").value;
var left = 1;
left = POSLEFT - left;
SETPOSX(left);
document.getElementById(namn).style.position="absolute";
document.getElementById(namn).style.left= POSLEFT;
document.getElementById("POSX").value= POSLEFT;
}

function Hoger(namn)
{
POSLEFT = document.getElementById("POSX").value;
var left = -1;
left = POSLEFT - left;
SETPOSX(left);
document.getElementById(namn).style.position="absolute";
document.getElementById(namn).style.left= POSLEFT;
document.getElementById("POSX").value= POSLEFT;
}

-->
</script>

```

*fig. 4.q – förflyttning av bild med JavaScript*

Denna kod placeras inne i head-taggen och anropas genom av en klickning på en av följande knappar.

```

<button name="Upp" onclick="Upp(<?PHP echo($typ) ;
?>);">Uppåt</button> <br>
<button name="Vanster" onclick="Vanster(<?PHP echo($typ)
; ?>);">Vänster</button>
<button name="Hoger" onclick="Hoger(<?PHP echo($typ) ;
?>);">Höger</button><br>
<button name="Ner" onclick="Ner(<?PHP echo($typ) ;
?>);">Ner</button>

```

*fig. 4.r – Anrop till förflyttning av bild*

Det som händer är att när någon klickar på knappen aktiveras attributet onclick i button taggen och kör den funktionen som står där. Den funktionen i sin tur aktiverar en annan funktion som flyttar bilden åt önskat håll. Jag implementerade denna funktionen i JavaScript och PHP för att administratören ska kunna få direkt visuell feedback och att plagget ska skapas utan behöva ha någon kunskap om HTML eller PHP.

Bildfilen laddas upp med samma bilduppladdningsfunktion som tidigare har beskrivits i rapporten och kommer där med inte ta upp den igen.

Val av typdel sker i en rullgardinslista genom att klicka på den typdel som man vill att den ska höra till.

När administratören lägger till en kläddetalj läggs en ny post in i databasen.

### **Uppdatera en kläddetalj (uppdatera.php)**

Redigeringen av ett klädesplagg fungerar på samma sätt som skapandet av kläddetaljen med undantag att man väljer vilken klädesdetalj som ska ändras.

### **Borttagning av kläddetalj (ta\_bort.php)**

Databasen hämtar först alla plagg till en rullgardinslista på administratören kan välja inom vilket plagg den önskade kläddetalj finns under. Nu listas alla kläddetaljer inom det plagget, genom att klicka på en specifik kläddetalj så tas den bort från databasen och den tillhörande bilden försvinner från webbservern.

### **Redigera lagda ordrar (uppdatera.php)**

Alla ordrar i databasen listas upp i en tabell där man kan ändra status och att granska en specifik order. När man granskar en specifik order hämtas all nödvändig information från databasen och skrivs ut på skärmen, en bild av klädesplagget representeras också.

Vid ändring av status uppdateras databasen efter önskat val. De olika tillstånden som går att välja för en order är: obehandlad, mottagen, levererad ej betald och levererad betald.

Tanken är att orderarna listas och administratören ska kunna gå i och ändra status på de olika orderarna vartefter produkten behandlas. Exempel: först lägger en kund en order på ett plagg, administratören går in och ser ordern och bockar för att han har sett ordern. Han skriver också ut all information som han behöver för att designern ska kunna se upp plagget. När designern har sytt upp plagget skickas plagget iväg och administratören loggar in och ändrar status på ordern till levererad ej betald. När kunden har betalt ordern ändras statusen till levererad betald.

Jag har valt att implementera det på detta sättet för att ge kunden respons vad som händer med plagget och att administratören ska hålla reda på vilka ordrar som han håller på med och hur långt det har gått med just en specifik order.

### **Borttagning av order (ta\_bort.php)**

Här listas också alla ordrar, genom att klicka på en text där det står "ta bort" tas ordern bort från databasen.

Vartefter kommer listan med ordrar att fyllas på och den behövs tömmas från inaktuella ordrar, därför har jag inplanterat denna funktion.

### 4.3. Användartester

Jag har haft fyra användartestare egentligen fem om man räknar med min handledare Nicholas Pagden som egentligen inte gjort någon användartest, men ändå gett feedback.

När jag valde användartestare ville jag ha representanter från bägge könen, i alla åldrar och olika bakgrund. Jag försökte välja folk som skulle kunna passa in i målgruppen för att få applikation målgruppsanpassad. Jag valde också folk som skulle kunna hitta buggar som har en viss vana av den här sortens programmering. Följande personer valde jag.

Användartestare 1 har nästan ingen datorvana men är van att sy kläder, hon är en äldre person.

Användartestare 2 är kläddesignern som jag gör examensarbetet för, hon har suttit med datorer men ändå är hon ingen expert på datorer.

Användartestare 3 är en person som studerar Webbprogrammering, hon är mer insatt hur webbplatser fungerar än att designa kläder.

Användartestare 4 är en kurskamrat som har god datorvana men ingen vana att designa kläder.

Den första iterationen av användartestet var redan vid det första prototypstadiet.

Jag skapade två olika prototyper (se stycket Prototyper under Utförandet för mer information om prototyperna och resultatet av testet). Detta test gjordes i tidigt i projektet och då vill jag mest ha ut vilken typ av applikation som jag ville skapa. Därför utformade jag först prototyperna och gav ut de olika prototyperna till testpersonerna som skulle testa prototyperna och svara på några enkla frågor, men främst ge egna kommentarer och åsikter. Frågorna som jag ställde handlade om användarvänlighet för kunden och om begränsningar av applikationen. Jag beskrev också hur jag hade tänkt hur de olika prototyperna skulle fungera.

När jag hade kommit en bit på vägen och fått de flesta funktionerna att fungera valde jag att göra ett nytt användartest.

Detta utförde jag på lite olika sätt.

Användartestare 1: satt jag ner med när jag utförde testet och lät henne registrera sig, designa ett plagg och lägga en order. Jag studera henne och hennes handlingar samt att jag ställde frågor under testets gång.

Användartestare 2: gjorde jag två iterationer med. För den första iterationen skrev jag ihop en manual och bad henne testa webbplatsen efter beskrivningar och bad henne om kommentarer. Den andra iterationen gick ut på att hitta buggar och fel.

Till användartestare 3 skrev jag ner frågor som handlade om användarvänlighet, enkelhet, buggar, funktion, navigation samt struktur.

Användartestare 4 använde en gammal mall om användartest som vi har fått på någon föreläsning. Mallen innehåller saker som man ska kolla på när man gör ett användartest samt frågor till användaren.

Sammanfattningen av det som jag kom fram till av användartestet var följande ting.

- Alla tyckte att det var för lite färg i gränssnittet och det var inte estetiskt tilltalande. (trots att jag påpekade att designern ville själv stå för gränssnittet av produkten)
- Använder fel termologi, exempel heter arm ärm i klädvärlden. Det fanns stavfel på sidan.
- JavaScripten fungerar inte i Mozilla Firefox.

- Den som har minst datorvana tyckte att det var lite komplicerat att designa ett plagg men förstod ändå hur hon skulle gå till väga för att designa plagget, de andra tyckte inte att det var svårt att designa sitt plagg.
- Problem med inloggningen, en person har haft problem med att logga in på både kund och administrations -delen.
- För mycket ”databaskänsla”.
- Textfälten är inte anpassat till texten som ska matas in.

Förutom en ren funktionstest frågade jag lite andra saker såsom de själva skulle tänka sig att köpa ett plagg på det här viset i framtiden?

De svarade enhälligt att de skulle tänka sig det om priset var rätt.

De återgärder som jag gjorde efter användartestet var att läsa på lite termologi inom klädbranschen, fixade till stavfelen, ändrade om designen av webbplatsens gränssnitt (trots att Helene Tegenfeldt vill skapa gränssnittet själv senare), Jag har inte brytt mig om att göra webbplatsen kompatibelt med Mozilla Firefox för att Internet Explorer har fortfarande flest användare, Jag har försökt komma ifrån ”databaskänslan” under administrationsdelen genom att flytta runt kläddetaljer visuellt. Storleken på textfält har också justerats.

En till användartest iteration är planerad och det är ska ske precis innan sidan tas i kommersiellt bruk.

## 5. Nästa skede

Det som måste göras innan produkten tas i bruk är att fundera på hur frakten ska vara, om det ska vara via posten eller någon annan expeditionsfirma. Om det ska vara fast eller rörligt fraktpris, om det är rörligt om det då ska vara baserat på vikten eller per kolli? Om det ska ske med postförskott, då tillkommer i dagsläget 50kr extra på fraktpriset. Detta beslut får de som kommer att driva webbplatsen att ta.

Bland det viktigaste när du bygger upp en Internetbutik är att skapa tillit. Detta för att kunden ska känna sig säker att inga personuppgifter läcker ut, transaktionen blir korrekt och att kunden får det som beställs och betalas för.

Detta kan man göra genom att skriva på webbplatsen att alla personuppgifter stannar inom företaget. Att transaktionerna kommer att ske via postförskott (Posten) eller genom något annat känt pålitligt företag. Det går också att återupprätta en så kallad secure server, detta ser kunden genom att det står <https://> framför adressen. Med en secure server kommer förmodligen kunden att känna sig tryggare.

Att ha en egen domän är viktigt att skapa tilltro till en webbutik. Alltså att adressen inte är <http://www.hem.passagen.se/knut456456/Knutswebbshop/index.htm> utan <http://www.designaklader.se>.

Ett problem kan vara att kundens designade plagg inte passar när han får plagget efter att designern har sytt upp det, plagget kan vara för litet eller för stort. Vad ska man göra åt det? Ett sätt är att i förhand skriva vad som menas med alla storlekar, om det är barn eller vuxenstorlekar, vad som händer ett plagg inte passar. Att göra upp ett kontrakt eller en text som kunden ser innan han beställer plagget.

Enligt Preece får man tilltro till en webbplats och en webbshop genom följande punkter.

- Genom att klargöra sammanhang i förnekanden eller interaktioner som finns. T.ex. personer eller företag kan föra fram bevis på vem de är, vilket kan vara i en form

av kompetens och tidigare framföranden i "business" eller ett uttalande av långsgående inblandning i ett Community.

- Skapa klara och sanningsenliga bekännelser, T.ex. Vi lovar att ta ansvar för fel och hjälpa till med problem.
- Känna till att tilltro/tillit är att ta en risk, ändå gör så att det baserat på gott rykte av kvalitet och pålitlighet.

"Trust is the expectation that arises within a Community of regular, honest, and cooperative behavior, based on Commonly shared norms, on the part of the members of the community, (fukuyama, 1995, p.x)"(Preece, 2001, s.191)

## 6. Sammanfattning och sluddiskussion

Systemet som jag har gjort är delvis begränsat, men jag har försökt komma ifrån mycket begränsningar genom att plaggen genereras efter administratörens alternativ.

Det som jag menar med delvis är begränsat är att det finns ett begränsat antal olika kombination av val (*se figur 3.b*) som kunden kan designa sina klädesplagg inom. Till exempel finns det bara eventuellt 3 olika sorters ärmar. Dessa begränsningar är det administratören som sätter.

Det jag har tänkt på när jag har gjort denna webbplats är att det ska vara lätt för både administratör och kund att skapa nya plagg men ändå att det inte ska vara begränsat. I framtida versioner av Macromedia Flash kan man kanske behandla genomskinliga bilder på ett vettigt sätt och då kan det vara lämpligt att göra en ny version av webbplatsen. Fördelen att göra det i Macromedia Flash är att det går att göra webbplatsen mer användarvänlighet och ett snyggare gränssnitt. Webbplatsen går också att bygga ut och släpa på, till exempel betalning med kreditkort, stöd för flera webbläsare, koppla ihop databasen med en expeditiönsfirma, osv. men detta är för eventuellt kommande versioner. Problemet som kan lösas i framtiden är att webbplatsen ska vara kompatibel med alla webbläsare då Internet Explorer tappar användare mer och mer.

Som slutsats kan jag säga att jag har skapat en webbplats där det är möjligt att designa sitt plagg för kunden och för en kläddesigner att sedan sy upp det om kläddesignerna matar in rätt information. Även om första syftet i små detaljer inte har utförts på det sättet som var meningen i början så tycker jag att jag har lyckats med uppgiften. De detaljer som inte har utförts som det är tänkt från början är att det var meningen att kunden skulle dra i punkter som utgjorde ett klädesplagg.

Jag har inte stött på några större problem under tidens gång förutom att Macromedia Flash inte hanterar externa genomskinliga bilder på ett bra sätt. Något som jag har upplevt bra nu efteråt är att jag inte började med att sätta mig direkt vid datorn och programmerade utan att den första tiden satt jag och tänkte igenom och skissade ner på hur jag skulle utforma webbplatsen. Alltså att jag hade ett långt planeringsstadium då jag tänkte igenom hur jag skulle utforma allt. Nu efteråt ser jag att det som jag planerade då stämmer överens väldigt bra med resultatet.

Lärdomar som jag kan dra av detta examensjobb är att man måste se på vad kunden vill ha. Att kunden kan vara moster Agda som nästan aldrig har suttit vid dator så programvaran ska vara så enkel att använda som möjligt och då får man eventuellt begränsa sin produkt. I detta fall var det svårt för en ej van kund att designa sitt plagg genom att dra i de olika punkterna, för kunden måste dels ha kunskap om hur ett klädesplagg är uppbyggt och dels hur man kan dra de olika punkterna. Då blir det lättare

att designa genom att göra olika val, det kan vara en bättre lösning trots att kundens del blir begränsad. Andra lärdomar är att man kanske måste tänka om och välja andra alternativ och plattformar, att kanske ta bort några krav som var från början för att göra produkten bättre. Alltså som uttrycket säger ”Kill your darling”.

Det har varit väldigt intressant att arbeta med detta projekt och jag har fått en mycket större insikt i PHP och JavaScript -programmering. Det som jag har lagt ner mest tid på är planeringsstadiet och skapa webbplatsen användarvänlig.

Till slut vill jag säga att detta projekt har varit väldigt lärorikt då man arbetar med ett riktigt projekt och åt någon. Det har också varit bra att arbeta själv då man har fått komma på alla lösningar själv, dock med lite hjälp från handledaren.

## Referenser

*100webspac* (2004) Members Control Panel (Elektroniskt)

Kräver ett användarkonto.

Tillgänglig <<http://freestarthost.com>>(2005-05-17)

Atkinson, Leanon. (2001) *core PHP programming Using PHP to Build Dynamic Web Sites 2<sup>nd</sup> ed.* Upper Saddle River: Prentice Hall PTR.

Buchanan George, Farrant Sarah, Jones Matt, Thimbleby Harold, Marsden Gary, Pazzani Michael. Improving Mobile Internet Usability

*International World Wide Web Conference archive*

*Proceedings of the tenth international conference on World Wide Web Hong Kong, Hong Kong, 2001, S.673 - 680.*

p673-buchanan.pdf

Tillgänglig

<<http://portal.acm.org/citation.cfm?id=372181&coll=ACM&dl=ACM&CFID=32859827&CFTOKEN=89134061>> (2004-11-24)

Fukuyama Francis. (1995) *Trust: the Social Virtues and the Creation of Prosperity.* New York: The Free Press, 1995

Hall, Johan, DAA751vt04 *Internetprogrammering* (2003). PHP - Hypertext Preprocessor (Elektroniskt) 3 skärmsidor.

Tillgänglig: <<http://student.msi.vxu.se:1234/index.php>>. DAA751vt04/Kursinnehåll /PHP & MySQL (2005-05-21)

inUse AB. (Elektroniskt)

*Riktlinjer för gränssnittsdesign* (2004), S.8-22.

Riktlinjer\_v1.pdf

Tillgänglig <[http://www.inuse.se/riktlinjer/Riktlinjer\\_v1.pdf](http://www.inuse.se/riktlinjer/Riktlinjer_v1.pdf)> (2005-05-30)

Jonsson, Viktor. (2001) *Webbprogrammering med PHP.* Lund: Studentlitteratur.

*Jumper.nu* (2004) Jumper.nu: Eget tryck (Elektroniskt) 2 skärmsidor.

Kräver Macromedia Flash Player 7.

Tillgänglig: <<http://www.jumper.nu/>> Produkter/Eget tryck. (2005-05-21)

*Körnefors, Rune, MEB715, Medieteknologi – Scriptprogrammering* (2005) MEB715 – Programmering (Elektroniskt) 8 skärmsidor.

Tillgänglig: <<http://w3.msi.vxu.se/multimedia/kurser/MEB715/>>. Kursinnehåll:/ 1 JavaScript. (2005-05-21).

Macromedia, Inc (Elektroniskt) 3 skärmsidor.

*Macromedia - Flash Player : At a Glance, 2001-2005*

Tillgänglig <<http://www.macromedia.com/software/flashplayer/productinfo/overview/>> (2005-05-30).

Michaelsson, Tony, *Webdesignskolan* (Elektroniskt) 28 skärmsidor.

*Webdesignskolan, CSS positionering*

Tillgänglig <[http://www.webdesignskolan.com/css\\_position/css\\_position.htm](http://www.webdesignskolan.com/css_position/css_position.htm)> (2005-05-29)

*PHP: Hypertext Preprocessor* (2005)(Elektronsikt)

Tillgänglig: < <http://www.php.net> >. (2005-05-21).

*PHPBuddy* (2005) *PHPBuddy.com – PHP vs ASP*(Elektronsikt)

Tillgänglig: < [http://www.phpbuddy.com/sub\\_articles.php?other\\_articles=9](http://www.phpbuddy.com/sub_articles.php?other_articles=9)> (2005-08-22)

Preece, J. (2000). *Online Communities: Designing Usability, Supporting Sociability*. West Sussex: John Wiley & Sons

*TrendEagle – KEPSDESIGNER*: (Elektroniskt) 1 skärmsida.

Kräver Macromedia Flash Player 6.0.

Tillgänglig: < <http://www.trendeagle.com/flash/index.asp> >.. (2005-05-21).

## Bilagor

### Date attribut.

De olika attributerna som kan används vi funktionen date(); är

a	Skriver ut om det är am eller pm
A	Skriver ut om det är AM eller PM
B	Swatch Beat time (En världsklocka som inte har några tidszoner)
d	Vilken dag det är i månaden, Med en inledande nolla
D	Vilken veckodag det är, Skrivet med 3 bokstäver
F	Namn på månad
h	Klockslag, (mellan 00-12)
H	Klockslag , (mellan 00-24)
g	Klockslag, (mellan 1-12, ingen extra nolla)
G	Klockslag , (mellan 0-23, ingen extra nolla)
i	Minuter
j	Vilken dag det är i månaden, Inte med inledande nolla
l	Veckodag
L	1 om det är skottår annars 0
m	Månadsnummer (0-12)
M	Vilken månad det är skrivet i bokstäver
n	Månadsnummer, utan inledande nolla
s	Sekund(00-59)
S	Suffix av vilken dag det är i månaden, (1st, 2nd osv)
t	Antal dagar i månaden
U	Sekunder sen epoken (tidsåldern).
y	År skrivet med två siffror
Y	År skrivet med fyra siffror.
z	Vilken dag det är på året (0-365)
Z	Tidzonsutjämning i sekunder

(Atkinson, 2001.:322-343)



Växjö  
University

Matematiska och systemtekniska institutionen  
SE-351 95 Växjö

Tel. +46 (0)470 70 80 00, fax +46 (0)470 840 04  
<http://www.vxu.se/msi/>