ARCADA

Scorify - Planeringen och utveckling av en alfaversion av en responsiv webbapplikation för poängsättning av spel

Jon Fredrik Ståhlström

Examensarbete

Informations- och medieteknik

2013

Jon Ståhlström

EXAMENSARBETE	
Arcada	
Utbildningsprogram:	Informations- och medieteknik
Identifikationsnummer:	4186
Författare:	Jon Ståhlström
Arbetets namn:	Scorify - Planeringen och utveckling av en alfaversion av en responsiv webbapplikation för poängsättning av spel
Handledare (Arcada):	Johnny Biström
Uppdragsgivare:	

Sammandrag:

Webbapplikationer har blivit allt mer vanliga och de har också blivit allt mer komplexa. Nya tekniska lösningar tillåter webbutvecklare skapa funktionalitet och användargränssnitt som närmar sig kvaliteten bland vanliga program och applikationer. Syftet med arbetet är att dokumentera processen för skapandet av en webbapplikation. Alla planeringssteg gås igenom och de valda lösningarna diskuteras. Meningen med detta är att visa åt läsaren hur en applikation planeras och vad som måste planeras före applikationen byggs. Efter planeringen skriver jag kort om utvecklingen av alfaversionen av applikationen. Applikationen i fråga är en applikation som sparar resultat från olika sporter och spel. Alfaversion kommer att ha ett nästan färdigt användargränssnitt och en del av den planerade funktionaliteten.

Nyckelord:	Webbapplikationer, HTML5, CSS3, Responsiv design, Scorify, MongoDB
Sidantal:	50
Språk:	Svenska
Datum för godkännande:	

DEGREE THESIS	
Arcada	
Degree Programme:	Information and Media
Identification number:	4186
Author:	Jon Ståhlström
Title:	Scorify - Planeringen och utveckling av en alfaversion av en responsiv webbapplikation för poängsättning av spel
Supervisor (Arcada):	
Commissioned by:	

Abstract:

Web applications are becoming more and more popular and their technical complexity is also increasing due to new technologies that are available to developers and designers. Web applications are quickly closing the quality gap that has existed between native applications and applications developed for the web. The aim of this thesis is to document the creation of a web application. The steps that are taken in the planning stage are discussed and the chosen solutions explainend. The goal is to give the reader an oversight over how a web application is planned before development is started. After the planning stage I will develop an alpha version of the application. The application in question is a scoring system for different games and sports. The alpha version will have some base functionality and the graphical user interface will be in a nearly finished stage.

Keywords:	Webapplications, HTML5, CSS3, Responsive design, Scorify, MongoDB
Number of pages:	50
Language:	Swedish
Date of acceptance:	

INNEHÅLL / CONTENTS

1 INTRODUKTION

Webbapplikationer har under de senaste åren blivit allt vanligare och mer komplexa, webbapplikationer kan redan i vissa fall jämföras med traditionella applikationer och program. En stor del av ökningen av webbapplikationer är HTML5 och CSS3 som tillåter webbutvecklare använda nya egenskaper i webbläsare och de apparater som webbläsaren är i. Snabbare Javascriptmotorer i webbläsare gör att man kan överföra vissa saker till klientsidan som förr har varit på serversidan. Det har också skapats flera nya Javascript ramverk som hjälper utvecklare bygga större applikationer med Javascript på klientsidan var serversidans kod kan minskas till endast kommunicering med databasen. Flera av de mobila operativsystemen gör reklam för att utvecklare kan skriva vanliga webbapplikationer till deras plattformer och med minimala ändringar till koden kan de användas som vanliga applikationer på telefonen.

1.1 Bakgrund

Jag fick idén till att bygga applikationen efter att jag hade använt flera olika applikationer för att spara mina resultat i frisbeegolf och andra sporter och spel. Alla applikationer som jag använde fungerade bra och var till nytta just då när jag behövde dem, men efter att jag hade använt dem en stund märkte jag att alla hade flera små problem och underligheter. Ett av de största problemen hade inte med applikationernas buggar att göra utan med det faktumet att jag måste ha en skild profil för varje applikation, oberoende om det var en webbapplikation eller en vanlig applikation för iPhone eller Windows Phone 7. Applikationerna varierade också mycket med det att få ut sitt data ur applikationen, några av dem lät en ladda ner en Excel-fil med viss statistik för alla spel eller endast för den senaste rundan/spelet. Efter att jag märkte att jag hade åtminstone ett par bekanta som hade liknande problem bestämde jag mig för att utveckla en webbapplikation som man skulle kunna spara resultat i vilken som helst idrottsgren och var man skulle få ladda ner sitt data och manipulera helt på de sätt som man vill.

1.2 Syfte och metoder

Syftet med detta arbete är att dokumentera planeringen och början av utvecklingen av en webbapplikation som låter användare spara resultat och föra statistik av flera olika spel och idrottsgrenar. Datat som applikationen sparar skall vara lätt att ladda ner från applikationen om användaren så vill. Applikationen skall fungera både mobilt och på vanliga webbläsare dvs den skall vara responsiv till skärmstorleken, mobilversionen kan vara avgränsad på några sätt men endast om det ger en bättre användarerfarenhet. I slutet av detta arbete skall applikationen vara färdigplanerad och möjligtvis ha basfunktionalitet som t.ex. inloggning och skapandet av användarprofiler klart, dvs applikationen är i slutet av arbetet en s.k. alfa-version av applikationen. Dessutom skall applikationen också vara i ett sånt skede att det är lätt att fortsätta utveckla den och implementera den funktionalitet som specificerats i planeringsskedet. Meningen är att läsaren får en överblick i de olika delarna som hör till planeringen av en webbapplikation och vilka arbetsskeden som hör till planeringsprocessen och till början av tekniska utvecklingen av applikationen.

I detta arbete använder jag mig av min egna erfarenhet inom branschen samt artiklar och en bok om responsiv design och webbutveckling. Jag använder också dokumentationen för de programmeringsspråken, verktygen och ramverken jag använder för att designa och bygga applikationen.

1.3 Avgränsning av arbetet

Jag utgår från att läsaren har baskunskaper i HTML och CSS och kan en del av den terminologi som används i webbutveckling och applikationsdesign. Jag avgränsar arbetet så att det är mest en rapport över planerings- och designprocessen för en webbapplikation. Jag börjar med en kort förklaring om responsiv design, efter det går jag kort igenom andra webbapplikationer och deras för- och nackdelar. Efter det börjar jag beskriva planeringsarbetet för applikationen och övergår sen till designen och början av den tekniska utvecklingen av applikationen.

1.4 Terminologi

Responsiv design – Design av webbsidor och webbapplikationer så att sidan skalas rätt i olika skärmupplösningar.

Back-end – Serversidan av applikationen, används för att hämta data från databaser och läsa filer och föra dem vidare till användarens webbläsare, klienten. Kan också innehålla sk. Business Logic för att formatera data före det skickas vidare till klienten

Front-end – Det grafiska användargränsnittet, samt klientsidans programmering.

HTML - Hypertext Markup Language, strukturen för webbsidor

CSS – Cascading Style Sheets, stilen för webbsidor

PHP – Hypertext Preprocessor. Programmeringsspråk, används på serversidan.

JavaScript – Programmeringsspråk, används oftast för att skapa interaktivitet i användargränssnittet och att hämta innehåll asynkront från en server.

Sidunderlag – (engelska: wireframe). Ramar för användargränssnittet, används för att visualisera hur applikationen beter sig då användaren använder den.

MySQL – Ett Relational Database Management System (RDBMS), Relationsdatabaser

MongoDB – Ett BSON-dokumentbaserat databassystem

XSS- Cross site scripting, ett datorrelaterat säkerhetsproblem, en sårbarhet i webbapplikationer som angripare kan utnyttja för att stjäla användares information.

jQuery – Ett Javascript bibliotek

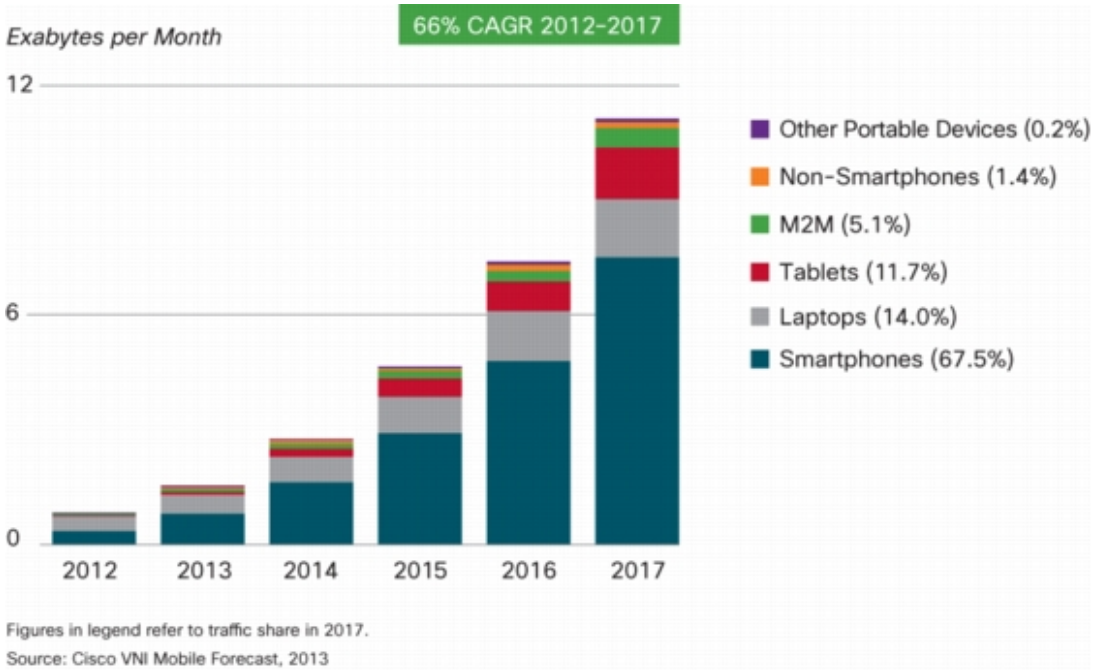
2 RESPONSIV DESIGN

I det här kapitlet hur termen responsiv design skapades, vad termen tekniskt sett betyder och hur jag kommer använda responsiv design i mitt arbete.

2.1 Termen responsiv webbdesign

Idéen för responsiv webbdesign började dyka upp på nätet runt 2009-2012 då det var klart

att mängden användare som läser webbsidor och använder webbapplikationer kommer att gå förbi den traditionella användningen på bordsdatorer och laptoppar (Meeker, 2010 | Ingram, 2010 | Datta, 2012). Därför måste också webbsidor och applikationer bli utvecklade på ett sånt sätt att de fungerar optimalt på alla plattformer (Wolski, 2013). Termen skapades av Ethan Marcotte i maj 2010 (Wikipedia, 2013d).



Figur 1, Internettrafik enligt plattform. (Cisco, 2013)

Den enklaste förklaringen för vad som menas med reponsiv design är att webbsidan eller applikationen ändrar utseende beroende på användarens upplösning på skärmen, allt innehåll finns på en webbsida och det formas om m.h.a. en CSS-fil.

Före responsiv design blev populärt brukade man oftast skapa skilda sidor för mobila apparater, oftast med "m" prefixet i webbadressen t.ex. http://m.dna.fi jämfört med: http://www.dna.fi. innehållet på dessa äldre mobilsidor är dock oftast optimerat endast till en skärmstorlek och innehållet fungerar inte då mobilsidan delas ut åt alla mobilanvändare oberoende av upplösningen eller skärmstorleken av apparaten de ser på sidan med, ett annat problem är att då användaren delar med sig sidan åt sina vänner kan det hända att nån kommer till mobilsidan med en vanlig webbläsare och mobilsidans layout och design ser inte bra ut i bordsdatorns webbläsare.

2.2 Teknisk förklaring

Responsiv design möjliggörs av CSS3 specifikationens `max` och `min-width` media-features som man kan implementera i `@media` media-queries kombinerat med en media-type t.ex. `all` eller `screen`. Det finns också andra media features, bl.a. `device-width`, `orientation`, `resolution` mm. De kan alla användas för att ännu noggrannare specifiera stilar för en viss plattform med vissa egenskaper, det är dock `max-` och `min-width` som används oftast då man skapar responsiva layouts i CSS eftersom de kombinerade med media-type `all` täcker en stor del av de plattformer med vilka användarna ser på sidorna eller applikationen. Regler för olika medier är dock inte helt nya i CSS3, de fanns redan i specifikationen för CSS2 men då specificerade man oftast endast olika stilar för "print" och "screen" dvs. hur innehållet ser ut då man printar en webbsida och då man ser på det med en skärm, dessa regler kallas för media-types (W3C, 2012). Detta löser inte problemet att allting med en skärm får samma stilar oberoende av upplösningen, man kunde delvis lösa det med att göra en så kallad flytande (engelska:"fluid") design och använda sig av procenter istället för bestämda pixelstorlekar för html-element, då formar sidan om sig automatiskt oberoende av skärmens upplösning men detta skapar också problem i användargränssnittet då upplösningen ändras. Två element som är 50 % breda på en 1000 px skärm kan se bra ut men när upplösningen ändras till 320 px blir elementen för smala och läsbarheten för innehållet lider. Samma sak händer då upplösningen blir för stor. Textraderna som är passligt långa då horisontella skärmens upplösningen är 1000px blir alltför långa och innehållets läsbarhet lider då skärmens upplösning blir större än 1000px. Designen kan också lida på andra sätt då man arbetar med en flytande layout, element som har rätt "tyngd" på en viss upplösning kan bli allt för stora och tar för mycket uppmärksamhet på mindre eller större upplösningar. I praktiken är det oftast menyer och andra interaktiva element som måste ha en viss tyngd som visuella element för att designen av applikationen skall fungera, det är också dessa element som oftast måste formas om beroende på skärmstorleken.

Med hjälp av CSS3 och media-queries kan webbutvecklare skapa flytande designer som formar om sig beroende på skärmens upplösning, ett exempel kan vara att en sida har två media queries: `@media all and (min-width:1024px)` och `@media all and (max-width:1023px)`. Efter den första media-query –regeln kan man bestämma att html elementet som har allt innehåll i sig är 1000 px brett och har 12px padding åt båda sidorna , då kan man använda procenter för att allt innehåll inne i det elementet fungerar och

flyter ihop rätt med andra element (Knight, 2011). Efter den andra media-query –regeln kan man bestämma elementets bredd till 480px, en vanlig upplösning för lite äldre smarttelefoner, då användaren kommer till sidan med en smarttelefon ser telefonens webbläsare regeln och använder de anpassade CSS regler som finns under den regeln även om de redan finns en gång i samma CSS–fil under den första regeln. På detta sätt kan webbutvecklare skapa en sida som laddar allt innehåll och sedan styra presentationen av innehållet till olika plattformer med CSS och media-queries, om man behöver skilda menyer eller gömma visst innehåll så går det lätt m.h.a. vanliga CSS–regler som `display:none` för det innehållet man vill gömma för mobila användare.

2.3 Responsiv design i mitt arbete

Jag kommer att designa applikationen så att den fungerar i både vanliga datorer, tablets och smarttelefoner, innehållet och utseendet kommer till en början vara samma på pekskärmsdatorer och vanliga bordsdatorer. Jag kommer rita sidunderlag för två olika skärmstorlekar, en smarttelefonversion och en pekskärm/bordsdator version. Pekskärmsversionen är så nära bordsdatorversionen att jag ansåg att det inte fanns värde i att rita skilda sidunderlag för båda. Då jag fortsätter utveckla applikationen kan jag i något skede skapa en skild version för högupplösningsskärmar och bordsdatorer, då skulle det finnas tre olika versioner av applikationen, en för mobiler, en för pekskärmsdatorer/bordsdatorer och en för bordsdatorer med högupplösningsskärmar dvs. skärmar med horisontell upplösning som är större än 2000px.

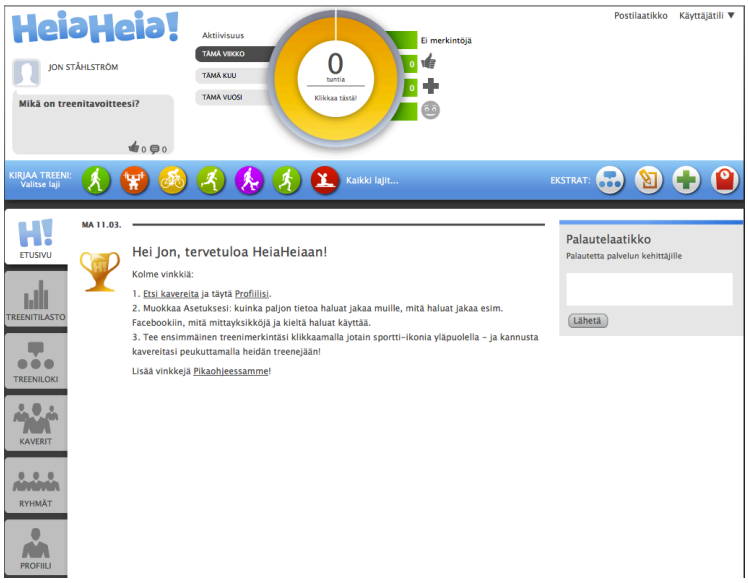
3 LIKNANDE APPLIKATIONER

För att få idéer för designen och funktionaliteten för applikationen gick jag igenom några applikationer som har lite liknande funktioner som det jag tänker skapa med scorify.

3.1 Applikationerna

1) HeiaHeia

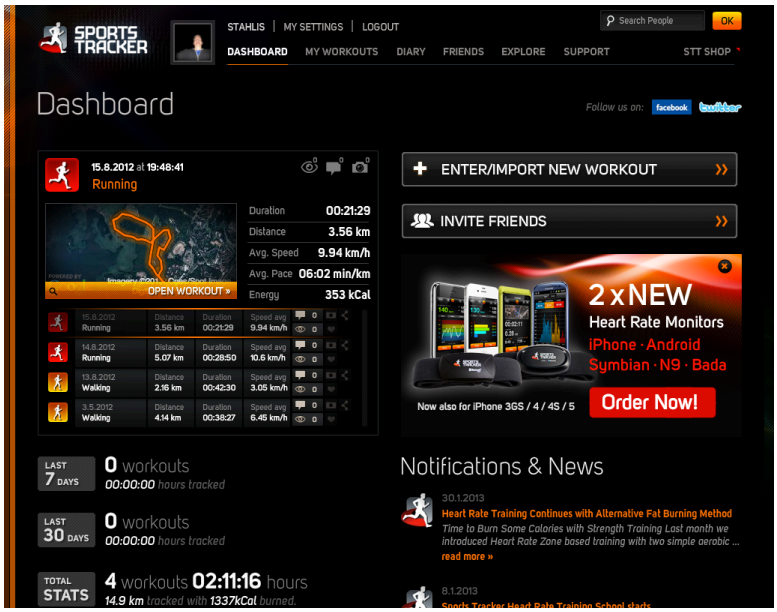
Applikationen sparar tider, sträckor och pulsmätares resultat för olika aktiviteter.



Figur 2, HeiaHeia.com, skärmbild.(HeiaHeia, 2013)

2) Sports Tracker

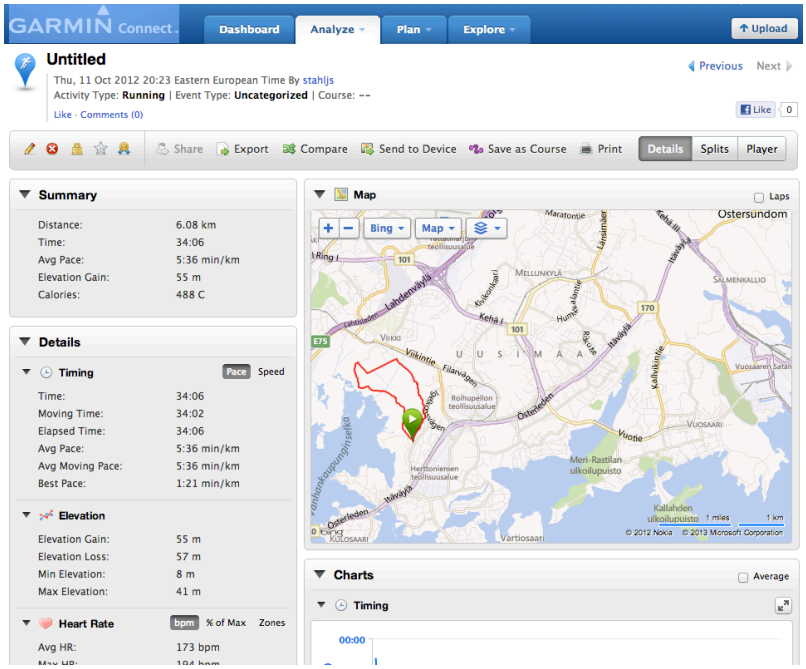
Applikationen sparar GPS och pulsmätarresultat från telefoner, klockor och dylikt. Har en webbapp som är byggd på Flash-teknologin, det finns mobila applikationer som kan laddas ner från: Nokia Store, Google Play, iOS Appstore och Windows Phone Store.



Figur 3, sports-tracker.com, skärmbild (Sports Tracker, 2013).

3) Garmin Connect

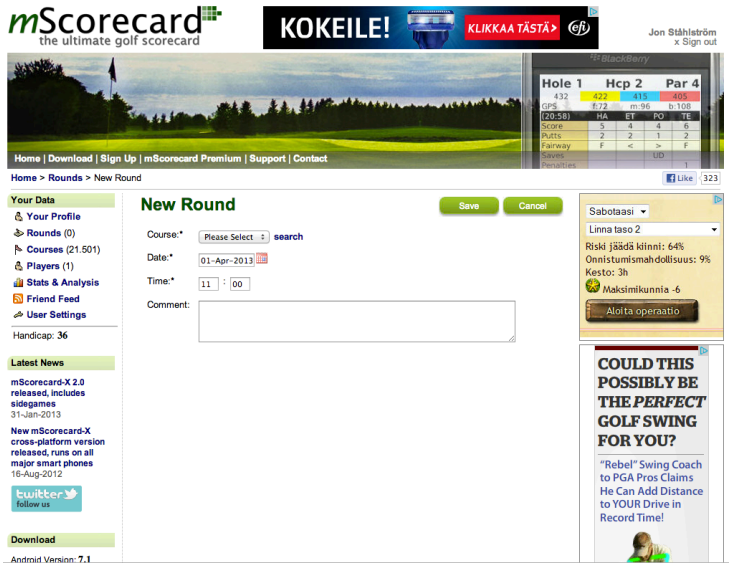
En webbapplikation som sparar data från Garmins GPS-klockor och pulsmätare, visar statistik om tider, sträckor, max och min pulser för användarens aktiviteter.



Figur 4, Garmin Connect, skärmbild (Garmin Connect, 2013)

4) mScorecard

En HTML5 applikation, gjord för att spara golfresultat och GPS data. Applikationen har också statistiksidor som visar gamla resultat.



Figur 5, mScorecard.com, skärmbild (mScorecar,2013)

3.2 Liknande applikationer, nackdelar och fördelar

De flesta applikationerna är inte ämnade för att spara poängresultat av idrottsgrenar, de är mer ämnade för att föra statistik över sträckor, tider och pulsmätningar. Den applikationen som har funktionalitet som liknar den funktionalitet jag söker till min applikation är det virtuella golfpoängkortet, mScorecard. mScorecard för statistik över de resultat man sparat och det går filtrera och sortera data i på statistiksidan på olika sätt.

Nackdelar för de flesta applikationerna är att de inte är responsivt byggda och i flera fall så kan det vara svårt och hitta informationen man söker i applikationen, t.ex. HeiaHeia har detta problem. En orsak till varför jag hade svårigheter med det var mängden reklamer som

fanns i HeiaHeia, mScorecard var bättre eftersom den hade lite mindre reklamer och menyerna var lättare att navigera.

HeiaHeia är den snabbaste applikationen med syn på att mata ett nytt resultat in på sidorna efter att man har registrerat sig som användare. Orsaken till detta är att applikationen har en knapp som är synlig på alla sidor i applikationen, knappen öppnar en valmeny för aktiviteten och efter det kan man mata in resultatet för den aktiviteten.

Det är klart att Sports Tracker är mer ämnat att användas på mobilen med just den applikationen som är ämnad för den plattformen man använder, webbsidan använder sig av Flash vilket gör att den inte är av det stabilaste laget och då jag försökte använda applikationen var det inte helt ovanligt för den att krascha. Sports Tracker har dock enligt mig den snyggaste designen av de applikationer jag tog som referenser, både webbapplikationen samt mobilapplikationen såg bra ut och hade bra uttänkta användargränssnitt.

Det var klart för mig att Garmin Connect endast är ämnat som en applikation att överföra data från Garmins produkter till en webbdatabas men jag tyckte ändå att applikationen hade många bra idéer och lösningar i hur det visade data som användaren laddat upp. Designen var också mycket enkel och klar och applikationen var lätt att använda.

mScorecard har enligt mig det bästa mobila gränssnittet, aplikationen är byggd med HTML5 och har ett mycket väldesignat gränssnitt, applikationen kan laddas ner som ett bokmärke på de flesta mobila plattformer och fungera då nästan som en vanlig mobilapplikation.

3.3 Tekniska lösningar

Sports Tracker var den enda av applikationerna som använder flash som plattform, de andra är normala webbapplikationer byggda på JavaScript/CSS och HTML. Heia Heia och mScorecard använder också jQuery ramverket för olika interaktiva delar i applikationerna. mScorecard använder också jQuery-UI ramverket som hjälper med olika saker i det grafiska

användargränssnittet. mScorecard gör också stort reklam för att de använder HTML5 för att skapa sin mobilapplikation.

4 FUNKTIONALITET OCH DATASTRUKTUR

I det här kapitlet går jag igenom planeringsskedet för applikationen, kapitlet beskriver hur jag planerade den funktionalitet applikationen skall ha, kapitlet beskriver också det data som applikationen behandlar. Jag går också igenom vilken databas jag använder för att spara data.

4.1 Funktionalitet

För att lättare kunna skapa sidunderlagen och en abstrakt datastruktur för applikationen började jag med att göra en lista över den funktionalitet jag ville att applikationen kommer ha. Listan jag gjorde av applikationens funktioner är en referens för fortsatt utveckling av applikationen, jag kommer inte under det här arbetet att utveckla en stor del av dessa funktioner. Listan av funktioner är inte i viktighetsordning utan i den ordning som jag skrev ner dem i planeringsskedet.

1) Skapa användare, normal registrering med användarnamn, lösenord och e-postadress. Fält för extra information som t.ex. riktiga namn och adresser skall finnas i med, men de är inte obligatoriska.

2) Registrera användare via Facebook eller Twitter, ger möjlighet för lättare delning av resultat och annan information. Applikationen kan hämta användarens information från Facebook eller Twitter för att underlätta registreringen av nya användare.

3) Applikationen skall ha vänlistor (t.ex Familj, Nära vänner) för användare, användare skall också kunna skapa nya listor och bli vänner med andra användare.

4) Användare skall kunna spara resultat i de idrottsgrenar eller spel som finns i systemet, om det är frågan om en lagsport skall man också kunna skapa nya lag eller välja

från listor med lag som redan finns sparade i applikationen. Användare kan vara med i lag som andra användare har skapat. Det skall vara möjligt att spara så noggranna resultat som möjligt. Till exempel i en ishockeymatch skall det vara möjligt att spara i vilken period, vilken tid och den spelarens namn i laget som gjorde ett mål. Det skall också vara möjligt att lämna fälten tomma och endast spara resultatet för hela matchen.

Tabell 1 *Exempel på detaljerad information om målen i en ishockeymatch*

Mål	Vilken period	Tid	Spelaren	Lagets namn	Annan informa ion	Powerplay eller inte?
Mål 1	1	19:39	Spelare1	Jons Lag		Jo
Mål 2	2	05:42	Spelare2	Jons Lag	Straffskott	

Tabell 2 *Exempel på kortforms data från en ishockeymatch*

Lagets namn	Hur många mål	Annan information
Jons lag	4	Vann på övertid
Roberts lag	3	----

5) Användaren skall kunna se detaljerad information/statistik om sina egna resultat och resultaten av de lag som finns i applikationens databas, om det är lag som andra användare har skapat skall den användare som skapat laget ha gett lov för att andra skall kunna se på statistiken för laget.

6) Användaren skall kunna få sitt data ut från applikationen dvs. applikationen måste ha någon sorts export –funktion.

7) En “skryta” funktion, knappar som delar vidare resultatet till Facebook, Twitter eller Google+. Inlägget på det valda sociala mediet visar resultatet från matchen/spelet och har en länk till applikationen. Om någon klickar på länken förs han till inloggningssidan för applikationen.

8) Turneringar och ligor. Användare skall kunna skapa ligor och turneringar för sina lag och bjuda andra lag med i ligan eller turneringen. Man skall också kunna be om att slippa med i turneringen eller ligan.

9) Meddelanden, om användarna skall kunna be att komma med i ligor skall det också vara möjligt för användare att skicka korta meddelanden och olika inbjudan till varandra. Meddelandefunktionen har olika nivåer beroende på om man är vän med den andra användaren eller om han endast är en som ordnar en större turnering eller liga.

4.2 Datastruktur och val av databas

Jag ville spara så noggrant data som möjligt från de olika spelen och jag ville att det är så enkelt som möjligt att öka på mängden fält i data samt att det är enkelt att ändra på databasens struktur. Dessa krav ledde till att jag valde MongoDB som databas för projektet.

4.2.1 MongoDB

MongoDB är ett NoSQL databassystem, NoSQL betyder att data inte sparas på samma sätt som relationsdatabas som MySQL och MSSQL. NoSQL databaser kan vara mindre komplicerade än traditionella relationsdatabas, det medför att de har bättre prestanda och passar väl för att snabbt kunna spara och söka stora set av data.(Kennedy, 2013) Eftersom de inte är relationsdatabaser går det inte att genomföra samma operationer på NoSQL databaser som på relationsdatabas, man kan använda liknande funktioner i MongoDB men i vissa fall måste man göra fler sökningar till databasservern för att få samma resultat som med relationsdatabas (MongoDb, 2013a). MongoDB sparar data i

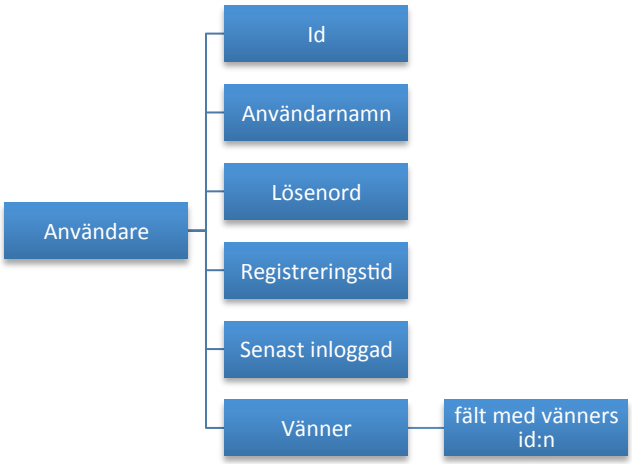
databaser som består av *collections*. *Collections* kan jämföras med tabeller från SQL-databaser. *Collections* innehåller dokument som är skilda set av nyckel:värde –par, dokumenten kan jämföras med raderna i en tabell i SQL databaser. Dokumenten är strukturerade som BSON –dokument, ett BSON dokument liknar JSON dokument men har några andra datatyper som kan sparas i dokumentet t.ex. en datatyp för att spara datum. (MongoDb, 2013b)

MongoDB började utvecklas 2007 av företaget 10gen och 2009 blev projektet öppen källkod. Då jag installerade MongoDB hade den senaste stabila versionen versionsnummer 2.2.3, denna version släpptes ut i Januari 2013.
Jag valde MongoDB för att jag hade hört om det på min arbetsplats och det passade exakt till de kraven jag hade för applikationen jag tänkte bygga. Jag läste också igenom flera artiklar som jämförde olika NoSQL lösningar och de flesta av artiklarna kom till slutsatsen att MongoDB var en av de mer mogna NoSQL lösningarna tack vare bra prestanda och funktionalitet.(Bushik, 2012 | Perfect Market, 2010)

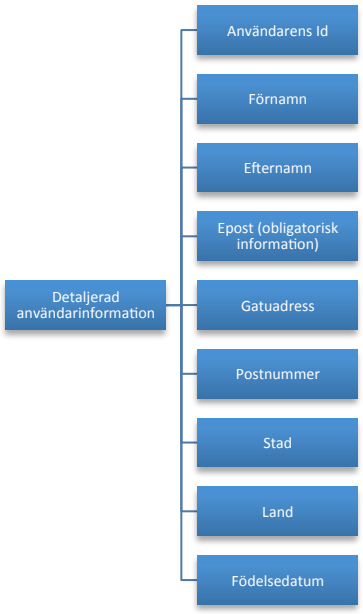
Jag behövde MongoDBs flexibilitet för att spara resultaten, användardata samt meddelanden. Data som applikationen sparar och söker kommer att ändras under applikationens utveckling och även efter att applikationen är klar, men med en flexibel databas slipper jag problem som jag skulle möta om jag använde ett traditionellt relationsdatabas. Problem som man kan förvänta sig är att man måste lägga till kolumner i en viss tabell och den ändringen kan det orsaka att man måste göra flera ändringar i koden av applikationen och om man inte är noggrann med var tabellens data används kan man lätt söndra applikationen samtidigt som man lägger till ny funktionalitet som t.ex. ett fält för att visa hur många som har tyckt om användarens lag på Facebook.

4.2.2 Datastrukturen

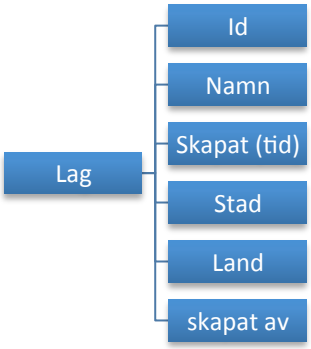
Jag skapade också en abstrakt struktur för data som jag vill spara från applikationen:



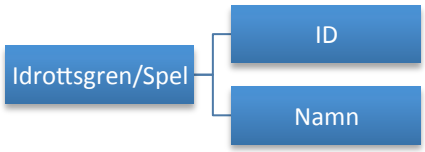
Figur 6, Struktur för användardata



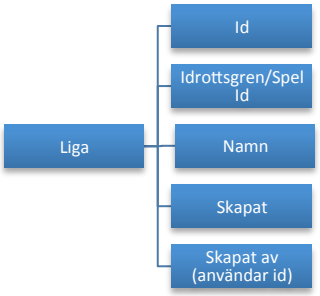
Figur 7, struktur för mer detaljerad användarinformation



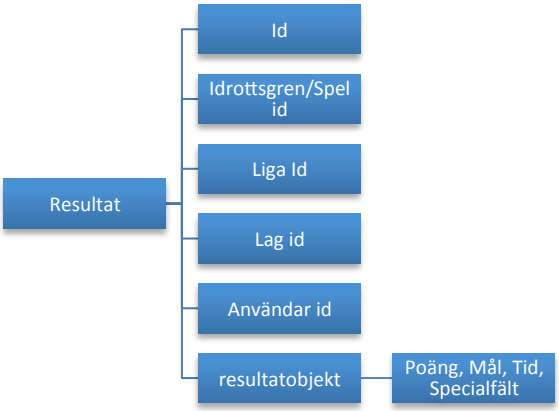
Figur 8, struktur för lagets data



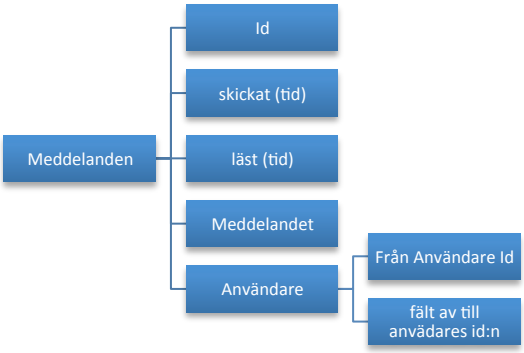
Figur 9, struktur för en idrottsgren eller ett spel



Figur 10, struktur för en liga, samma struktur kan användas för turneringar



Figur 11, strukturen för ett resultat, resultatobjektet i figuren ändras beroende på data som sparas.



Figur 12, strukturen för användarnas meddelanden.

4.3 Teknisk planering

Jag valde att skriva serversidans kod med PHP eftersom det är programmeringsspråket jag har mest erfarenhet med. Man kan skriva serversidans kod i nästan vilket som helst programmeringsspråk, enda kravet för denna applikation var att språket hade drivrutiner och stöd för MongoDB databasen.

För klientsidans använde jag vanlig HTML, CSS och JavaScript. För att underlätta JavaScript programmeringen valde jag att också använda jQuery JavaScript-biblioteket. I designskedet och när jag bygger det grafiska användargränssnittet kan jag använda flera andra hjälpbibliotek för jQuery.

5 UTSEENDE OCH ANVÄNDARGRÄNSSNITT

En av de viktigare delarna i en modern applikation är användargränssnittet eftersom det är det första och oftast det enda användaren lägger märke till, datastruktur och funktionalitet är inte mycket värda om de inte presenteras åt användaren på ett sånt sätt att informationen i applikationen är lätt att hitta och läsa. Som exempel kan tas menyerna i applikationen, menyerna skall vara lätta att navigera och skall inte kräva att användaren måste spendera mer än en kort stund för att lära sig att hitta det han söker efter i applikationen

Utseendemässigt kan man med hjälp av design styra användarens öga till de funktioner och den information som man vill att har i huvudrollen i den delen av applikationen. Sätt att styra användaren är genom ikoner, färger och olika storleks typsnitt. (Grannell, 2010 | Cannon, 2012)

Man måste också gå noggrant igenom vem som är i målgruppen för applikationen och vilka förkunskaper man förväntar sig att personen som använder applikationen har (Troeth, 2013). En receptapplikation för 40+ mänskor kan kräva ett helt annorlunda användargränssnitt än en receptapplikation för yngre, på samma vis kan en applikation som är ämnad för kockar se annorlunda ut än en applikation ämnad för amatörkockar

Man måste ta alla dessa saker i beaktande då man utvecklar applikationens utseende och användargränssnitt. Man måste också komma ihåg att designa en applikation som väcker en viss sorts känsla i användaren för att användaren skall komma ihåg applikationen och förhoppningsvis nämna den åt sina vänner.

5.1 Sidkartan

Jag började med att rita upp en sidkarta för applikationen, sidkartan fungerar lite liknande som diagrammen för datastrukturen i och med att den visar vilka sidor som skall finnas i applikationen och vilka sidor är under andra sidor. Sidkartan hjälpte också när jag började rita sidunderlagen, sidkartan visade vilka sidor behövde sidunderlag.

Jag gjorde sidkartan med tanke på första versionen av applikationen, den första versionen av applikationen har inte all funktionalitet som är planerad och därmed har sidkartan inte heller alla sidor som kommer finnas då jag vidareutvecklar applikationen.

Sidkartan:

1. inloggningssidan
    a. Ny användare
    b. Glömt lösenord
    c. Nytt lösenord för användare
2. Framsidan för användaren
    a. Användarens information
    b. All statistik
        i. Lagstatistik
        ii. Ligastatistik
        iii. Grenstatistik
    c. Nytt resultat
        i. Välj idrotssgren
        ii. Välj liga

iii. Lägg in resultat

3. Vänner och meddelanden

i. Vänlista

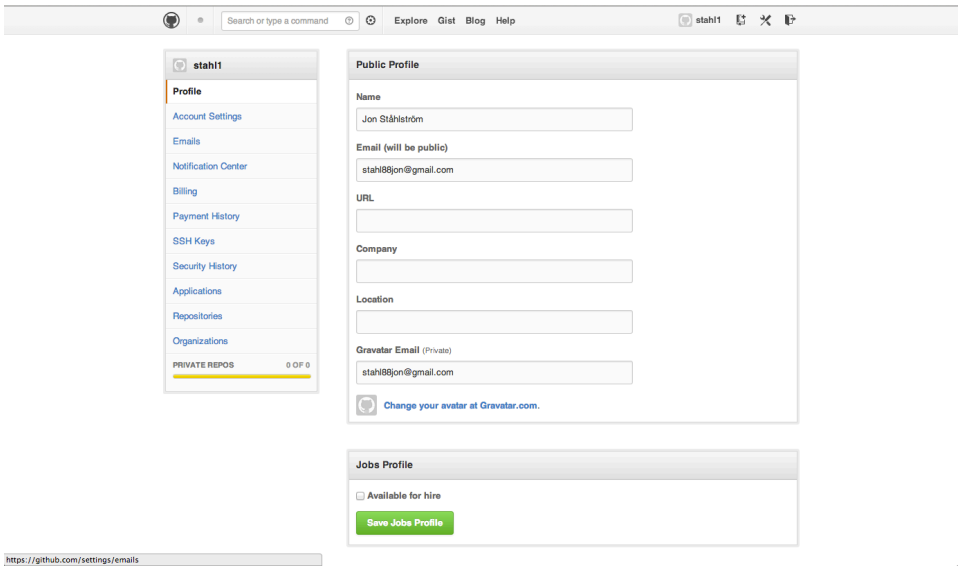
ii. Meddelanden

5.2 Referenser

Före jag började rita sidunderlagen och designa applikationens utseende samlade jag en lista med referenser och gick igenom de delar jag tyckte fungerade och de designelement som gjorde ett intryck på mig. Saker som jag speciellt lade märke till var positionering och storlek av menyerna, storleken av logon, val av typsnitt och på hurudan rutnät applikationen var gjord på.

1. Github

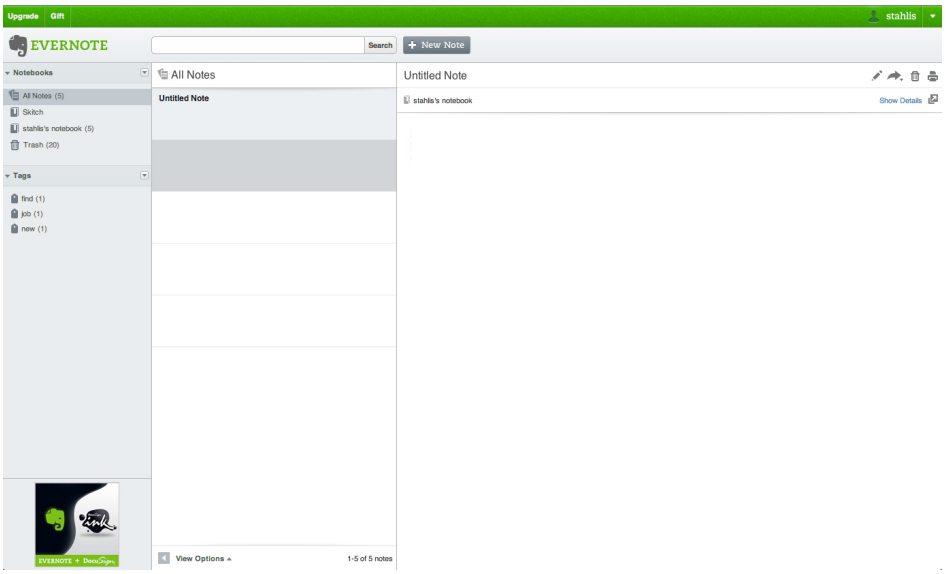
Jag tyckte Github var ett bra exempel på en sida som klarar av att vara sparsam med färger men har ändå ett klart utseende som man kommer ihåg och förknippar med sidan. Huvudmenyn är lätt att navigera och har endast de nödvändigaste länkarna i den. Github har också valt att använda ikoner för de navigationselement som för användaren till sina inställningar och användarinformation samt skapandet av ett nytt lagerutrymme (engelska: repository). Länkar som är som text i huvudmenyn för användaren till andra lagerutrymmen och till hjälpsidorna för Github.



Figur 13, Inställningssidan på Github (Github, 2013)

2. Evernote

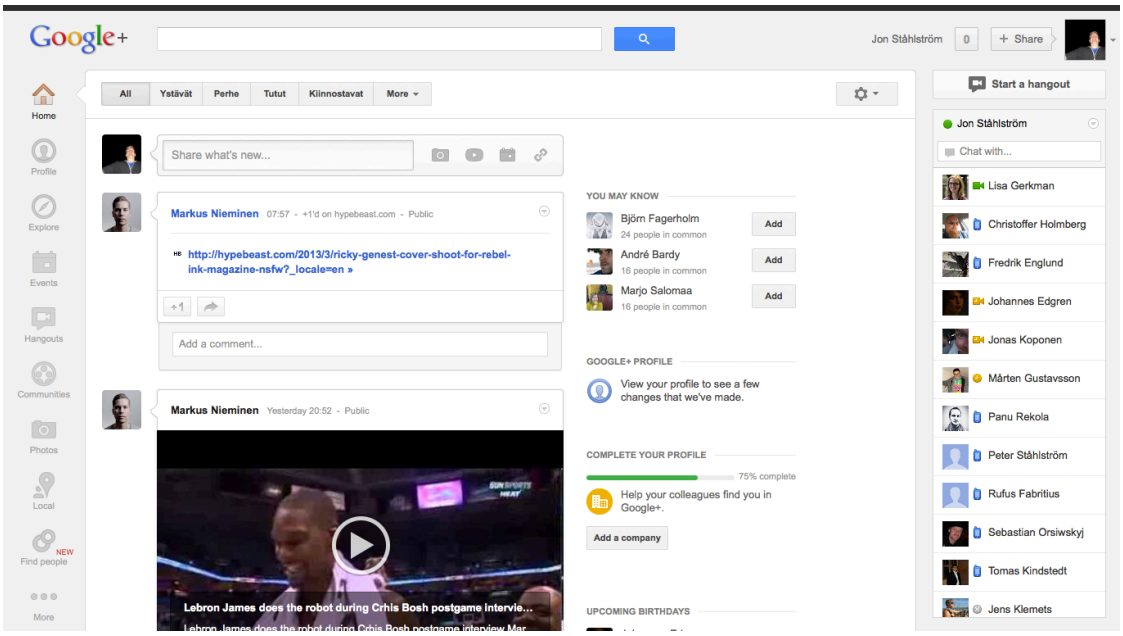
Jag tyckte om hur Evernote delar upp navigationen i två delar, en balk högst uppe på sidan som har länkar till användarens information och länkar för uppgradering till ett betal konto. Själva applikationens funktionalitet är på vänstra sidan av sidan. Evernote har också en väldefinierad färgvärld med tre klara huvudfärger.

10) 

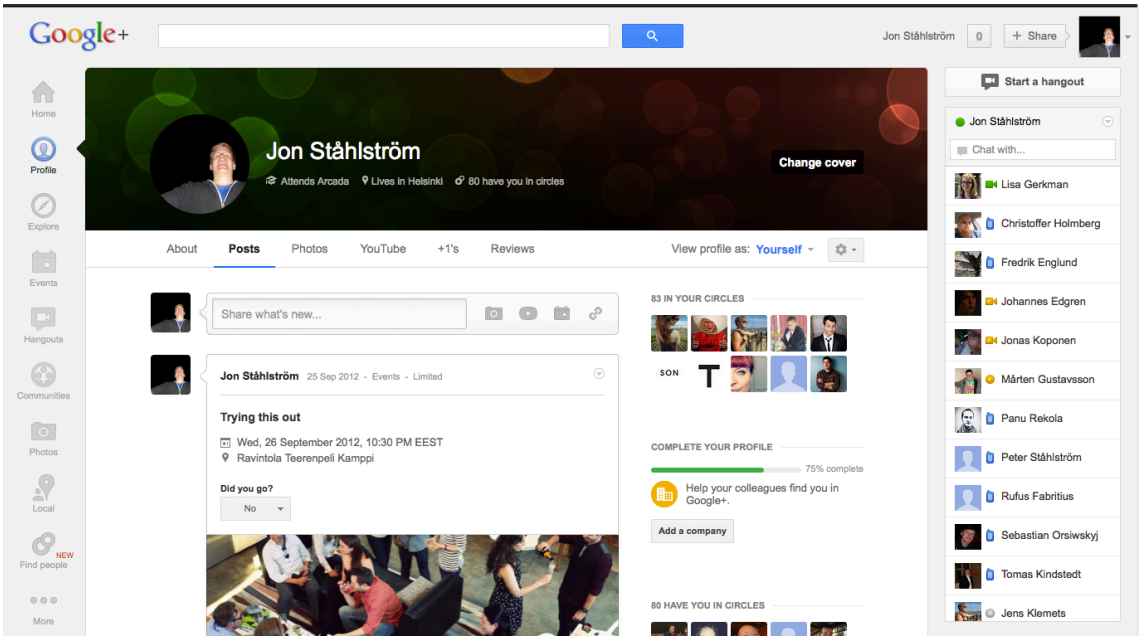
Figur 14, Skärmbild, användargränssnittet för Evernote (Evernote, 2013).

3. Google+

Google+ var närmast den sortens användargränssnitt jag sökte att implementera i min applikation, navigationen är lätt och förstå och på ett bra ställe på vänstra sidan av sidorna. Varje undersida kan vid behov ha en egen meny högst uppe på sidan. Jag tyckte också om det att Google valt att både ha ikoner samt text för varje länk i huvudmenyn. Nyhetsflödet är också gjort så att man kan lätt skilja de olika inläggen.



Figur 15, Skärmbild Google+ nyhetsflöde, (Google+,2013)



Figur 16, Skärmbild Google+ profilsida, (Google+,2013)

Från de tre exempelsidorna tog jag idéer som jag använde i planeringen av applikationens utseende. Menyn för applikationen är på vänstra sidan men menyn för en enskild användares personuppgifter och profilinställningar är i en liten balk högst uppe i applikationen. Innehållet i mitten byts och det kan finnas olika menyer i mitten som är relaterade till just det innehållet t.ex. sorteringsmöjligheter på statistiksidorna.

5.3 Sidunderlag för applikationen

5.3.1 Vad är Sidunderlag?

Sidunderlag fungerar delvis som en grund för applikationens utseende men deras viktigaste funktion är att på ett visuellt sätt visa vilket innehåll kommer vart i applikationen och vilka delar som behövs i användargränssnittet (Kelway, 2009 | Usability.gov, 2013 | Wikipedia, 2013c). Sidunderlagen hjälper också designern eller projektledaren att uppfatta vilka steg användaren tar då han/hon använder applikationen, fel i navigationen och brister i hur applikationen fungerar märks oftast i det skedet när man ritar upp sidunderlagen för applikationen. Det finns flera olika sätt att rita sidunderlag, sidunderlag kan vara ritade på papper för fri hand i början av planeringen och kan sedan övergå till mer detaljerade sidunderlag som är ritade med ett program som är mer ämnat för ändamålet. (Goltz, 2013). Man kan också skriva kommentarer i sidunderlagen och på det sättet kan t.ex. projektledare förklara visuellt det beteende som de vill ha i applikationen åt utvecklaren om han inte har varit med i det planeringsmötet var just den funktionaliteten eller strukturen av applikationen har skapats. För grafiker kan sidunderlagen fungera ramar för designen av sidorna, det kräver dock att man har skapat mer detaljerade sidunderlag som är speciellt ämnade för att underlätta designarbetet.

5.3.2 Ritande av sidunderlag

Jag brukar vanligtvis rita sidunderlag för varje vy i applikationen men för denna applikation prövade jag med en annan metod, jag ritade endast de vyer som var absolut nödvändiga och som hade funktionalitet som delas mellan olika sidor, t.ex. sidan var användaren ändrar sin profilinformation liknar sidan för inmatning av nya resultat. Sidorna för inmatning av resultat kommer också för det flesta vara olika sorters formulär beroende på idrottsgren och den information man vill spara. Statistiksidorna är tabeller med data vilka varierar beroende där också på vilken idrottsgren det är frågan om. Jag ritade också ut de vyer var jag var osäker om funktionaliteten i den delen av applikationen och där jag var osäker om hur användargränssnittet skulle vara format.

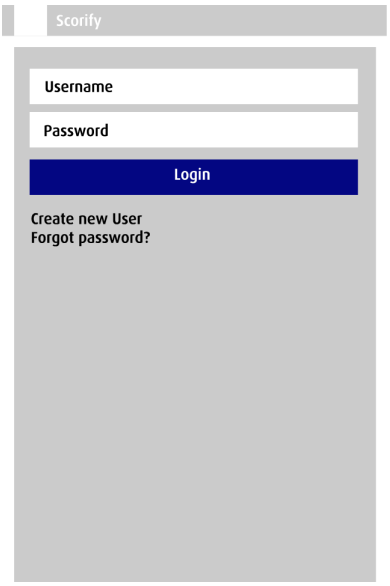
Jag använde Adobe Photoshop CS 5.5 för att rita sidunderlagen, som tidigare nämndes så finns det flera program som är särskilt gjorda för att göra sidunderlag men eftersom jag hade

erfarenhet med Photoshop, ansåg jag att tiden som skulle gå till att lära sig ett nytt program var bättre att använda på att planera och rita sidunderlagen.

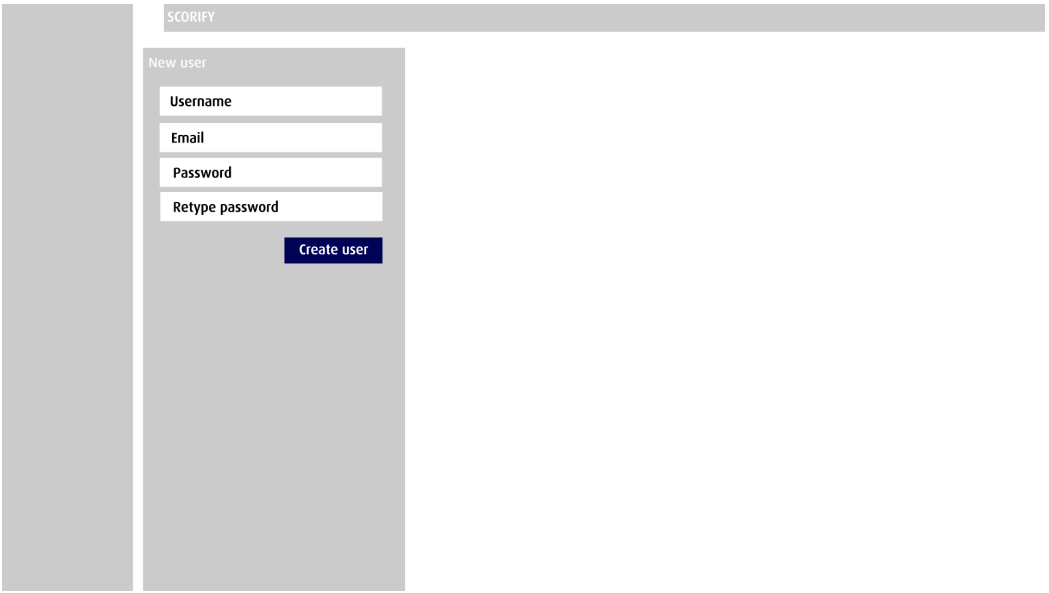
Att rita sidunderlagen var en av de svårare delarna av planeringen och designen av applikationen, detta för att jag försökte göra dem så noggranna som möjligt för att sedan kunna använda dem som riktlinjer för hur den egentliga designen av applikationen skulle se ut. I och med att jag gjorde lite mer detaljerade sidunderlag skapade jag också måtten för flera av elementen på sidorna t.ex. sidomenyn på vänstra sidan i applikationen är 240px bred på pekskärmsdatorer och bordsdatorer.

Sidunderlagen:

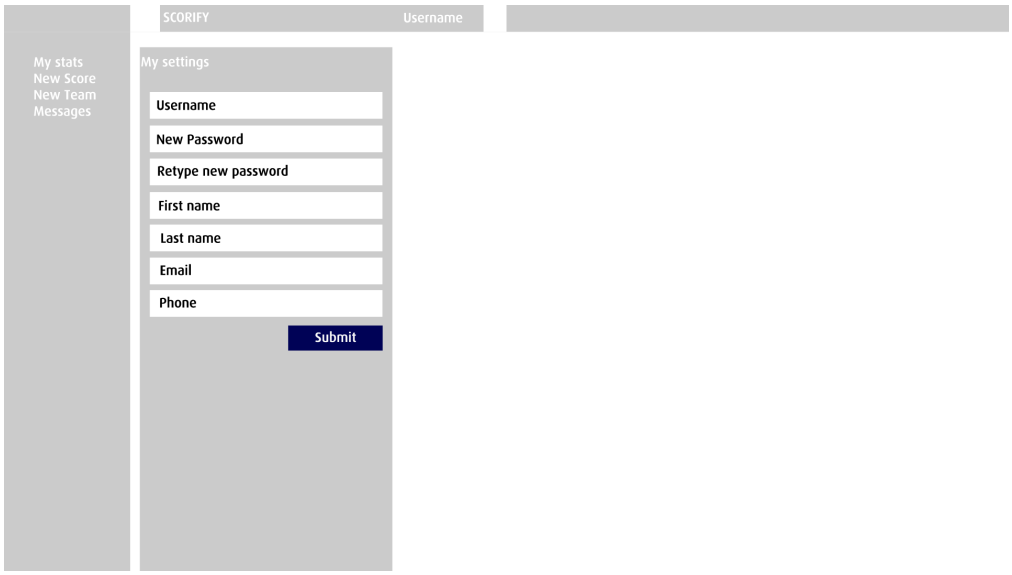
Figur 17, Inloggningssidans sidunderlag



Figur 18, Mobila login sidans sidundrelag

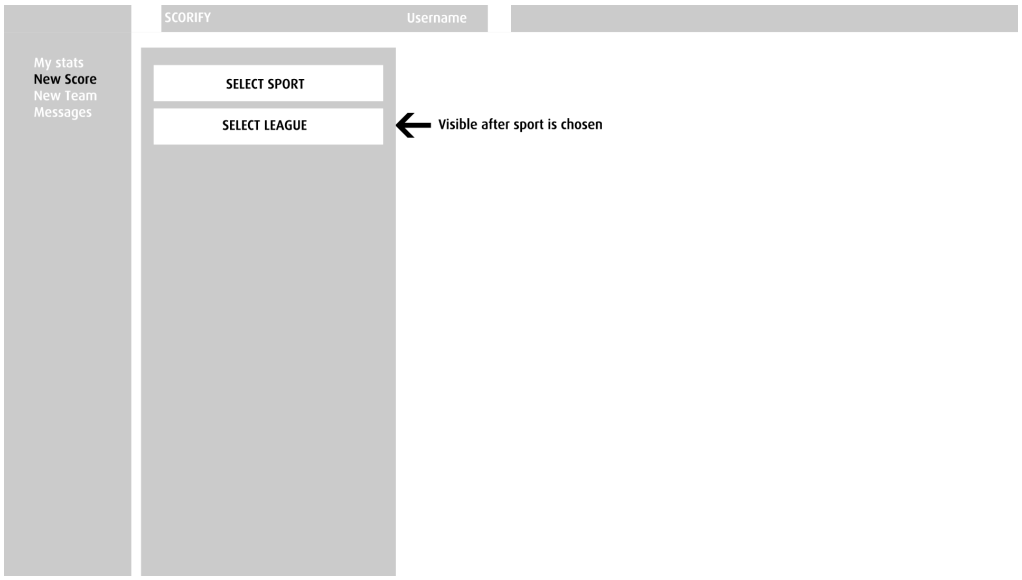


Figur 19, Skapa ny användare -sidans sidunderlag



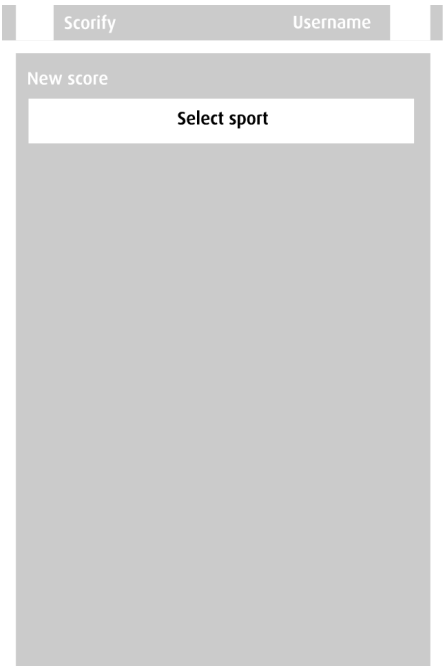
Figur 20, Inställnigssidans sidunderlag

Figur 21, Statistiksidans sidunderlag



Figur 22, Sidunderlag, Nytt resultat steg1

Figur 23, Sidunderlag, nytt resultat steg2

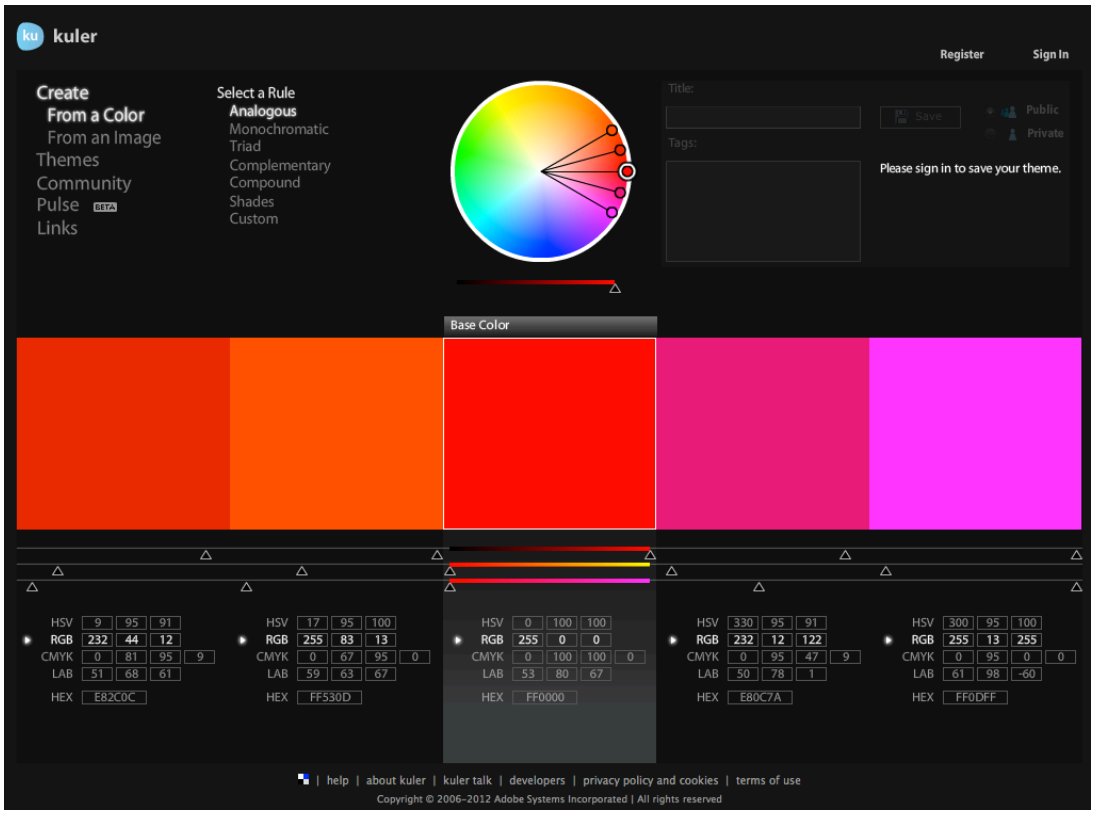


Figur 24, Sidunderlag, mobilt nytt resultat steg1

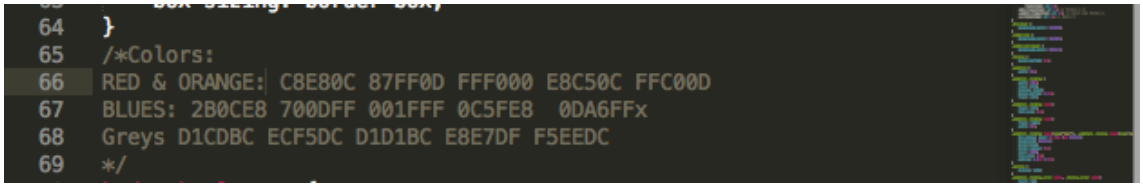
5.3.3 Färger

Efter att sidunderlagen var färdiga fortsatte jag med att välja färger för applikationen, färgvalen är ett sätt att styra användarens öga till de områden på sidan som man vill ha fokus på, man åstadkommer detta m.h.a. kontraster mellan olika färgade områden och text med en viss färg på en viss färgs bakgrund. Färgerna kan också skapa känslor i användare, ljusare färger är mer energiska medan mörkare färger ger ett intryck av lugn (Cannon, 2012). Skillnaden mellan dessa känslor är viktiga då sidorna eller applikationen har olika ändamål, t.ex. statistiksidan i min applikation kunde ha en mörkare bakgrundsfärg för att göra det lättare för användaren att läsa graferna och mängden av information på sidan. På alla sidor var det finns ett formulär kunde applikationen ha ljusare färger för att skapa uppmärksamhet för de olika textfälten och valmöjligheterna.

Jag valde färgvärlden med Adobe Kuler (Adobe Kuler, 2013), Kuler är ett online verktyg med vilken man kan skapa olika teman d.v.s samlingar av 5st färger. Man kan också använda teman som andra användare har skapat, jag valde att skapa flera olika teman, ett med olika varianter av grått, ett med olika blåa och ett med gula och orange färger.



Figur 25. Skärmbild av Adobe Kuler, (Adobe Kuler, 2013)



Figur 26, Färgvalen insatta i CSS -filen

5.3.4 Typsnitt

Jag valde typsnitten för applikationen med tanke på att de måste fungera i alla webbläsare, detta ledde till att ett val skulle ha varit att använda systemtypsnitt som finns på de flesta plattformer som t.ex. Arial. Problemet är ändå att typsnitt kan ha olika teckenmellanrum beroende på operativsystemet och webbläsaren, det påverkar utseendet då en text som annars ryms på en rad hamnar på två rader eller bryter ut från containern den skall rymmas i (Stackoverflow, 2011). Jag valde därför att hämta typsnitten från Googles WebFonts tjänst (Google Webfonts, 2013) för att typsnitten blir då endast påverkade av webbläsarens aliasering och t.ex. teckenmellanrummet påverkas inte lika mycket av plattformen. Jag valde att inte hosta typsnitten själv eftersom om användaren redan besökt sidor med samma typsnitt

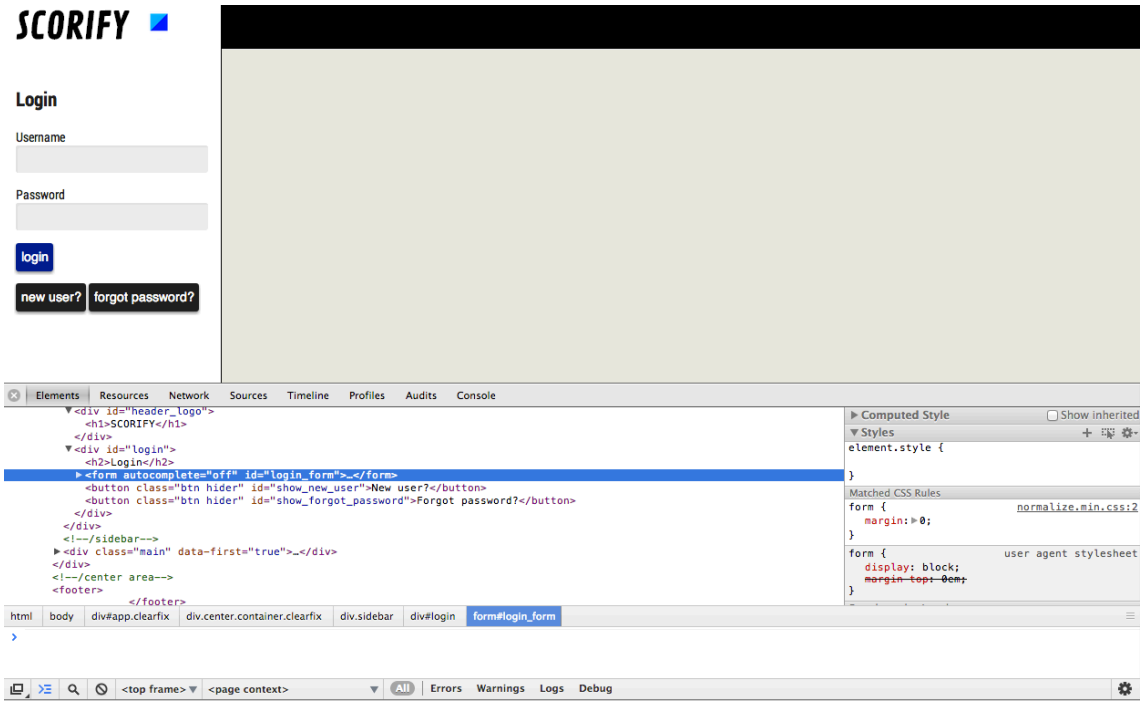
som jag har som är hostade på Googles servrar så sparas det tid vid sidladdningen då användarens webbläsare inte behöver söka typsnittena på nytt.

Jag valde typsnittet Roboto Condensed för brödtexten i applikationen. Roboto är ett sansserif typsnitt och fungerar bra i både vanliga webbläsare och mobila gränssnittet, typsnittet skapades för Android operativsystemet. Det är utvecklat för högupplösningsskärmar och menyer i användargränssnitt (Google, 2013b). Som text för logon och som typsnitt för specialfall valde jag Contrail One för att jag tyckte att typsnittet passade bra ihop med Roboto men var ändå tillräckligt personlig för att fungera som specialtypsnitt i logon för applikationen. Contrail är skapat av Riccardo De Franceschi och är utvecklat för bruk på webben som ett typsnitt för rubriker (Google, 2013a).

5.4 Design i webbläsaren

Det vanligaste sättet att designa webbsidor och applikationer är att en grafiker skapar designen för sidan/applikationen i Photoshop och ger den vidare åt en utvecklare som sedan överför designen till HTML och CSS. Man kan också designa rakt i webbläsaren, då börjar man genast med att bygga HTML och CSS, först bygger man HTML strukturen för sidan och sedan börjar man skapa designen med CSS. När designen utvecklas ökar också mängden HTML element på sidorna. Man kan jämföra Photoshops penslar och fill funktioner med en CSS fils kod som ger färger åt typsnitt och bakgrundsfärger åt HTML-blockelement. Själva HTML: en kan jämföras med Photoshops objekt och former. Fördelen med att skapa designen för applikationen med webbläsaren och en texteditor (engelska: Designing in the browser) är att man skapar den tekniska delen av användargränssnittet på samma gång som man designar, man undviker också problem som kan uppstå då man designar i Photoshop och skapar detaljer som inte är möjliga att göra med HTML och CSS. Man kan också lättare visa hur interaktiva element fungerar och snabbt ändra på designen för dem. Nackdelen med att designa i webbläsaren är att det tar lite längre att göra designprototyper med HTML och CSS än att rita designprototyper med Photoshop. De flesta webbläsare har i dagens läge inspektionsverktyg med vilka man kan rakt påverka HTML och CSS och se ändringarna ske i realtid i webbläsaren.

Jag valde att designa i webbläsaren, jag hade sidunderlagen som bas för designen, färgerna och typsnittena var också redan valda så jag började med att skapa skelettet för applikationen i HTML och sedan välja färger och storlekar för elementen i CSS filen.



Figur 27, Google Chrome Inspector

5.4.1 HTML5 Boilerplate

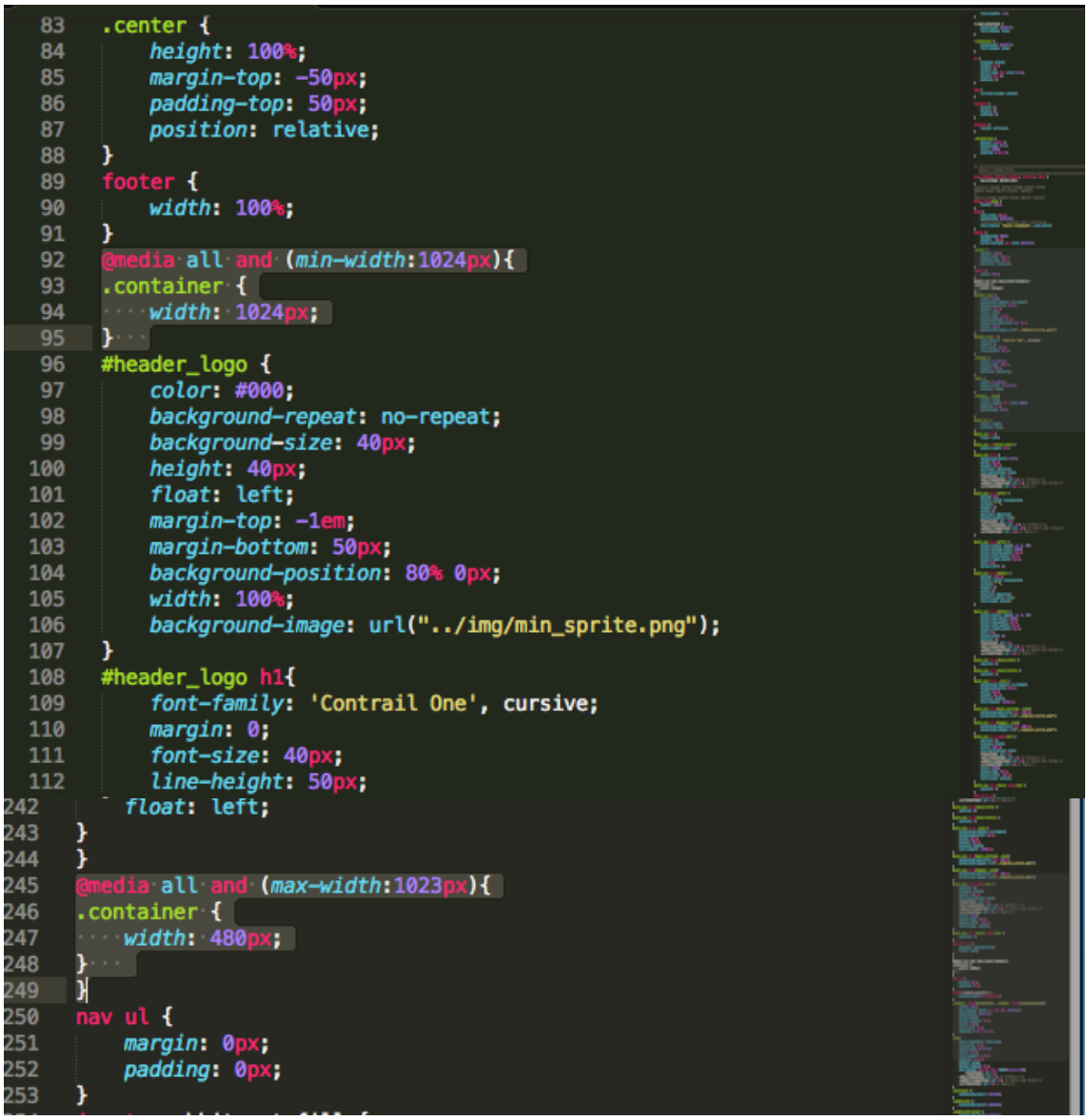
För att snabba upp utvecklandet i webbläsaren använde jag mig av HTML5 Boilerplate (HTML5 Boilerplate, 2013), ett färdigt paket med de mest använda JavaScript biblioteken och en CSS-reset fil. CSS-reset filen nollställer webbläsarens egna stilar för HTML elementen och gör så att de beter sig likadant i alla webbläsare. JavaScript biblioteken som kommer med i paketet är jQuery (jQuery, 2013) och Modernizr (Modernizr, 2013). jQuery är ett bibliotek som underlättar JavaScript utveckling genom att skapa funktioner för animering och modifiering av HTML element. Man kan skriva egna bibliotek som gör samma sak som jQuery men oftast är snabbare att använda det som finns färdigt, men det finns specialfall då prestandan av en viss del av applikationen är kritisk och då kan man inte använda sig av jQuery utan måste bygga en egen funktion. Modernizer är ett bibliotek som gör att äldre webbläsare som t.ex. Internet Explorer 8 får stöd för HTML5 element, även om de inte ur-

sprungligen stödde dem. Jag valde att ta bort Modernizer från min installation eftersom min applikation riktar sig på endast webbläsare som har inbyggt stöd för HTML5 och CSS3.

5.4.2 Val av brytpunkten för den responsiva designen

Jag började med att i CSS filen definiera de två brytpunkterna i den responsiva designen, en är för skärmar som är minst 1024 px breda, jag valde upplösningen för att det är upplösningen på flera peksskärmsdatorer och det är också en upplösning för flera gamla datorskärmar och det är storleken som är vanlig då man har ett webbläsarfönster som inte är i fullskärmsläge. För mobila layouten blev då brytpunkten högst 1023 px breda skärmar. Bredden för mobila innehållet är 480 px, 480 px är standardupplösningen för Windows Phone 7 telefoner.(Wikipedia, 2013b) Äldre Androidtelefoner och iPhone 3g och 3gs skalar sidan rätt då webbläsaren i dessa telefoner simulerar ett fönster som är 480 px brett om man använder rätt meta-taggar i headern för HTML-filen.

Jag valde att sätta containerns (HTML elementet som har alla andra element inom sig) bredd till den samma som brytpunktens upplösning. Containerbredden är det som styr alla andra lådors storlek eftersom jag programmerade deras bredder som procenter av föräldrablocket. Detta leder till att om containern är 1024 px bred så är sidomenyn 23.4375 % bred. Jag räknade ut procenten genom att dela ett pixelvärde som jag ville ha för sidobalken, 240 px med containerns bredd 1024 px. (240/1024) *100 = 23.4375 % .



```
.center {
    height: 100%;
    margin-top: -50px;
    padding-top: 50px;
    position: relative;
}
footer {
    width: 100%;
}
@media all and (min-width:1024px){
.container {
    width: 1024px;
}
#header_logo {
    color: #000;
    background-repeat: no-repeat;
    background-size: 40px;
    height: 40px;
    float: left;
    margin-top: -1em;
    margin-bottom: 50px;
    background-position: 80% 0px;
    width: 100%;
    background-image: url("../img/min_sprite.png");
}
#header_logo h1{
    font-family: 'Contrail One', cursive;
    margin: 0;
    font-size: 40px;
    line-height: 50px;
    float: left;
}
}
@media all and (max-width:1023px){
.container {
    width: 480px;
}
}
nav ul {
    margin: 0px;
    padding: 0px;
}
```

Figur 28, media-queries för brytpunkterna

5.4.3 Bildelement, ikoner och logo

Efter att jag hade skapat en prototyp för designen med de nödvändigaste elementen började jag designa ikonerna och logon för applikationen, för detta måste jag använda Photoshop. Jag gjorde ikonerna med hjälp av olika vektorformer och sparade dem i en .PNG fil som en s.k. sprite, om alla ikoner är sparade som en bild måste webbläsaren göra endast en filhämtning och detta kan snabba upp sidladdningen avsevärt. (Wikipedia, 2013b). Jag sparade bilden i .PNG formatet för behålla transparensen i bilden.

5.4.4 Verktyg i designarbetet

För HTML, CSS och JavaScript programmering använde jag programmet Sublime Text 2, Sublime Text visar innehållet för en mapp i samma fönster och har bra kodifyllning för HTML, CSS och JavaScript. Webbläsaren jag använde för designen var Google Chrome Version 26.0 beta. För vidare testning av sidan använde jag också Firefox version 19.0.2 med Firebug instickningsmodul installerad. Firebug är också ett inspektionsprogram som fungerar nästan på samma sätt som inspektören i Google Chrome.

6 TEKNISK UTVECKLING

Det här kapitlet omfattar den tekniska utvecklingen av applikationens alfaversion. Skapandet av databasen och programmeringen av grunderna till serversidan av applikationen.

6.1 Alfaversionen

För att skapa alfaversionen bestämde jag mig för att utveckla funktionaliteten för inloggningen, skapandet av nya användare, nollställandet av lösenord och inmatningen av ett resultat från ett videospel. Samtidigt som jag byggde på funktionaliteten fortsatte jag med designen av applikationen.

6.2 Databasen

För att kunna använda MongoDB måste man installera både MongoDB servern samt drivrutinerna för det programmeringsspråk man använder för att tala med MongoDB servern. Jag installerade MongoDB på Mac OSX enligt instruktionerna på deras sidor och drivrutinen för PHP. Efter att jag testade att MongoDB servern fungerade och att PHP drivrutinerna var av rätt version installerade jag ett PHP- baserad databashanteringssystem phpMoAdmin, så att jag hade ett grafiskt användargränssnitt med vilken jag kunde komma åt MongoDB servern.

Efter detta skapade jag databasen för applikationen och de *collections* som jag behövde för att bygga alfaversionen av programmet. Dessa *collections* var för användarinformationen,

ligorna, spelen och lagen.

6.3 Inloggningen och användarinformationen

Säker inloggning är en viktig del av alla webbapplikationer som har den funktionaliteten, lösenorden skall sparas säkert i en databas och det skall hashas och saltas för minska risken för att lösenordet skall knäckas och att användarens information kan bli sårbar. (Peslyak, 2010) Inloggningen skall också helst ske över en skyddad förbindelse (SSL/HTTPS) för att minska risken att någon snappar upp användarens data.

Jag valde att använda ett färdigt skript, phpass för att säkra lösenorden som jag sparar i databasen. Phpass används av flera CMS: ar b.la WordPress och har flera funktioner för att hasha lösenord på det säkraste sättet som är möjligt, beroende på den versionen av PHP som skriptet körs på. (Openwall, 2013).

Jag sparade användarens inloggning i en sessionsvariabel som föråldras efter ett dygn. Inloggningen, skapandet av nya användare och editerandet av användarinformation sker med AJAX, i alfa versionen av applikationen byggde jag endast kontroller för innehållet på klientsidan (tomma fält o dyl.), serversidan antar att innehållet finns och rensar endast bort saker som kan leda till XSS attacker dvs. HTML taggar tas bort. Jag hade inte tillgång till en server med ett SSL certifikat så i utvecklingsskedet skickas data osäkrat över från klientsidan till servern. Detta är inte ett problem så länge jag utvecklar den lokalt och jag är den enda användaren.

6.4 Inmatning av resultat

Jag valde att spara spelet Pong in i databasen och utveckla en inmatningssida för resultat från spelet. Inmatningssidan fungerar som inloggningssidan med AJAX, formuläret har två vyer beroende på om användaren vill spara tidpunkterna för poängen i spelet. På serversidan sparas data in i databasen på två olika sätt beroende på om det är detaljerad poängdata eller enkelt resultat med två värden för poängen.

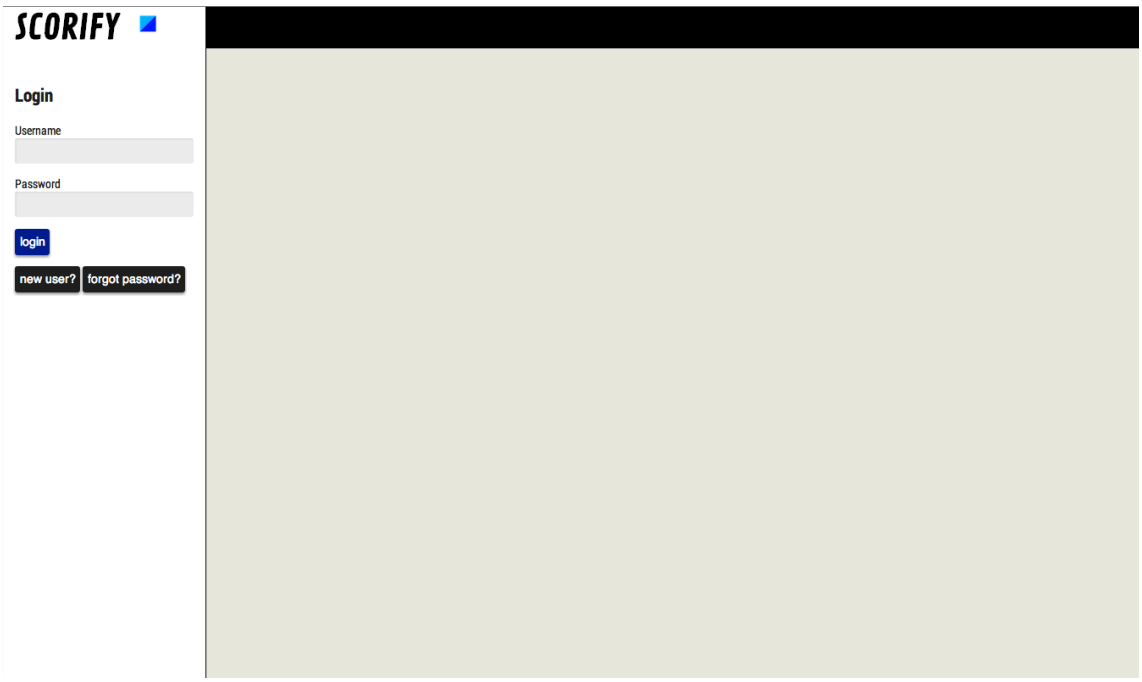
6.5 Verktyg i utvecklingsarbetet

Jag fortsatte utveckla applikationen med Sublime Text 2 och Google Chrome, för att skapa spelet och ett par lag i databasen använde jag phpMoAdmin som grafiskt användargränssnitt.

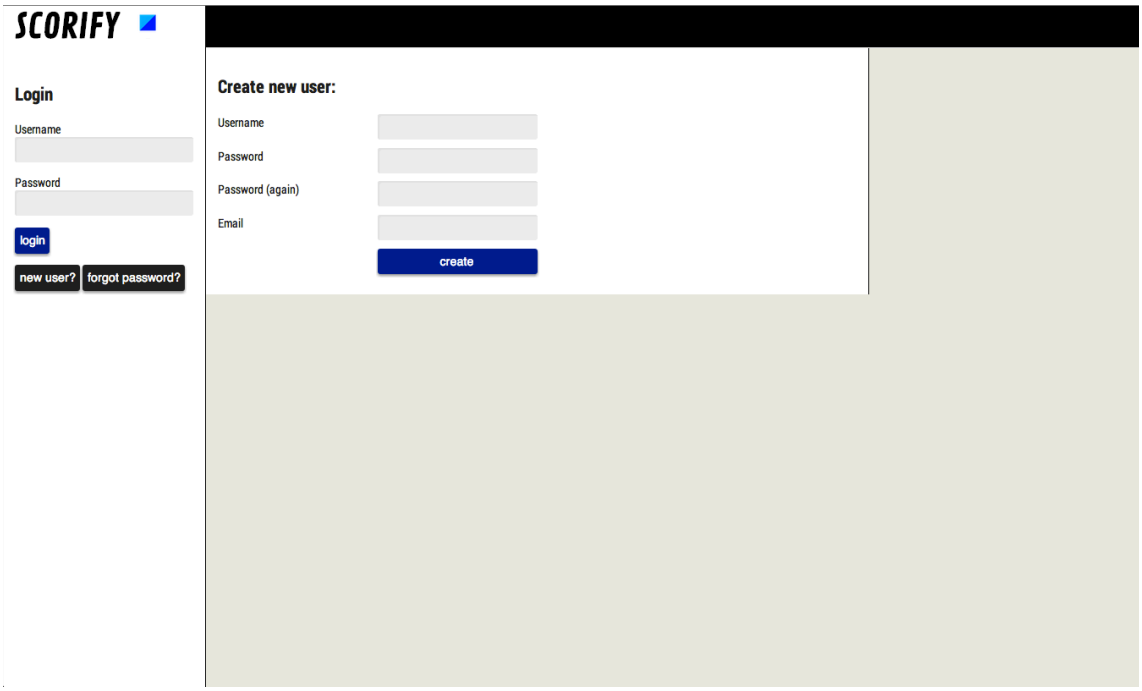
7 RESULTAT OCH REFLEKTION

7.1 Resultat

Planeringsarbetet är färdigt för applikationen, sidunderlagen är ritade och en grundläggande sidkarta är gjord. Applikationen är nästan färdigt designad och har ikoner för de element som finns i alfaversionen. Användargränssnittet fungerar och har en bra grund som är lätt att fortsätta bygga på. Funktionaliteten i applikationen är en del av det som planerades under arbetets lopp, användargränssnittet fungerar som det skall och det finns en databas för applikationen. Databasen är för applikationen kommer att flyttas inom snar framtid till en server och adressen för alfaversionen kommer att vara: http://www.servern.fi/scorifyalpha.



Figur 29, Scorify, inloggningssidan



Figur 30, Scorify, skapa ny användare

7.2 Reflektion

Det främsta målet för arbetet var att visa vilka steg som krävs i processen för att bygga en webbapplikation. Flera av delarna i arbetet kan tillämpas till byggandet av vanliga applikationer och webbplatser. Då man utvecklar en applikation från början märker man hur många arbetsskeden som krävs före man kan ens tänka om att börja programmera applikationen och implementera den funktionalitet man vill att applikationen skall ha.

Jag var personligen överraskad över hur invecklad planeringsprocessen blev, jag hade förvisso förväntat mig att det inte skulle vara alltför lätt att planera och designa applikationen men varje gång jag hade skapat en lista över funktionalitet som jag ville att applikationen skulle ha kom jag på nånting som var fel med den eller något som fattades. En annan sak jag märkte i planeringsskedet var hur tidskrävande sökandet av goda referenser för både funktionaliteten och designen var.

För mig var designen det mest intressanta i arbetet, troligtvis för att jag inte tidigare har gjort designen för några större projekt utan endast för små projekt eller som en del av ett

team som designat en sida. Jag hade förväntat mig mindre regler för designen av en applikation men jag förstod snabbt att för att kunna bygga en bra applikation måste designen lyda vissa regler som gör att t.ex. användargränssnittet fungerar på ett visst sätt.

Själva programmeringen av applikationen gick ganska smärtfritt och var en av de lättaste delarna av arbetet, detta kan vara för att det är just det jag har jobbat med inom branschen men det verkade som om allting gick snabbt framåt och med mycket färre problem än jag hade väntat mig. De problem jag stötte på var förknippade med MongoDB och installationen av det på Mac OSX. Problemen med installationen uppstod då jag försökte installera drivrutinen för PHP. Jag hade installerat MAMP paketet, MAMP paketet kommer med flera olika versioner av PHP och varje version hade en egen konfigurationsfil var man måste skriva in att man använder MongoDB utvidgningen. Man måste också se till att man installerar utvidgningen i rätt mapp beroende på den version av PHP man använder. Det visade sig också att den drivrutinen jag laddade ner inte stödde den versionen av PHP som var aktiverad i MAMP på min dator. Det fanns dock en inofficiell version av drivrutinen som fungerade med den versionen av PHP som MAMP använde.

Alfaversionen av applikationen är inte slipad och det finns mycket kvar att bygga på den men jag är nöjd med den lilla funktionalitet som jag hann utveckla. Jag tänker fortsätta utveckla applikationen och har satt som ett mål att få en betaversion klar före hösten, jag har också ansett att det är lättare om jag tar med flera och jobba med projektet för att snabba upp arbetet. Flera medarbetare kan också lättare märka fel i applikationen som jag som ensam utvecklare kan lätt bli blind till.

Detta arbete har alltså visat kort början av en webbapplikation med alla steg i arbetsprocessen synliga. Jag tror man kan få olika saker ur arbetet beroende på om man är en grafisk designer som läser arbetet eller om man är en utvecklare eller om man är en projektledare. Arbetet kan alltså hjälpa i att visa åt dessa olika delar i en arbetskedja hur deras kolleger arbetar och hur de kan möjligtvis hjälpa varann för att optimera deras arbetsprocesser och skapa bättre produkter snabbare och pålitligare.

KÄLLOR

Marcotte, Ethan. *RESPONSIVE WEB DESIGN*. A Book Apart, New York 2011. 157s. ISBN 978-0-9844425-7-7

Meeker, Mary. 2010. *Morgan Stanley Internet Trends, April 12, 2010.* Tillgänglig: http://www.scribd.com/doc/29850507/internet-Trends-Mary-Meeker-04-12-2010 Hämtat 15.3.2013

Cisco Systems, Inc. 2013. *Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2012–2017.*
Tillgänglig:
http://www.cisco.com/en/US/solutions/collateral/ns341/ns525/ns537/ns705/ns827/white_paper_c11-520862.html Hämtat 16.3.2013

W3C. 2012. *Media Queries W3C Recommendation 19 June 2012,* Tillgänglig: http://www.w3.org/TR/css3-mediaqueries/ Hämtat 17.3.2013

Peslyak, Alexander. 2010. *How to manage a PHP application's users and passwords.* Tillgänglig: http://www.openwall.com/articles/PHP-Users-Passwords Hämtat 24.3.2013

Openwall. 2013. *Portable PHP password hashing framework*. Tillgänglig: http://www.openwall.com/phpass/ Hämtat 24.3.2013

MongoDB. 2013a. *Data Modeling Considerations for MongoDB Applications.*
Tillgänglig: http://docs.mongodb.org/manual/core/data-modeling/ Hämtat 17.3.2013

MongoDB. 2013b. *Introduction to MongoDB*. Tillgänglig:
http://www.mongodb.org/about/introduction/ Hämtat 17.3.2013

Kennedy, Michael. 2013. *The NoSQL Movement, LINQ, and MongoDB - Oh My! - DevelopMentor.* Tillgänglig: http://www.develop.com/mongodb Hämtat 17.3.2013

Bushik, Sergey. 2012. *A vendor-independent comparison of NoSQL databases: Cassandra, HBase, MongoDB, Riak.* Tillgänglig: http://www.networkworld.com/news/tech/2012/102212-nosql-263595.html?page=6 Hämtat 19.3.2013

Perfect Market. 2010. *NoSQL Solution: Evaluation and Comparison: MongoDB vs Redis, Tokyo Cabinet, and Berkeley DB [CHART]*. Tillgänglig: http://perfectmarket.com/nosql-solution-evaluation-and-comparison-mongodb-vs-redis-tokyo-cabinet-and-berkeley-db-chart/ Hämtat 19.3.2013

Grannell, Craig. 2010. *Testing UX design IN DEPTH Experts demystify the process behind UX design.* Tillgänglig: http://www.techradar.com/news/internet/the-web-designer-s-guide-to-user-experience-658868/2#articleContent Hämtat 20.3.2013

Knight, Kayla. 2011. *Responsive Web Design: What It Is and How To Use It.* Tillgänglig: http://coding.smashingmagazine.com/2011/01/12/guidelines-for-responsive-web-design/ Hämtat 17.3.2013

Datta, Ashish. 2012. *Be Responsive: A History of Responsive Design.* Tillgänglig: http://shout.setfive.com/2012/03/12/be-responsive-a-history-of-responsive-design/ Hämtat 15.3.2013

Troeth, Stephanie. 2013. *Designing For The Multifaceted User.* Tillgänglig: http://uxdesign.smashingmagazine.com/2013/03/12/design-multifaceted-user/ Hämtat 20.3.2013

Ingram, Mathew. 2010. *Mary Meeker: Mobile Internet Will Soon Overtake Fixed Internet*. Tillgänglig: http://gigaom.com/2010/04/12/mary-meeker-mobile-internet-will-soon-overtake-fixed-internet/ Hämtat 14.3.2013

Kelway, James. 2009. *The what, when and why of wireframes.* Tillgänglig: http://userpathways.com/2008/06/the-what-when-and-why-of-wireframes/ Hämtat 21.3.2013

Usability.gov. 2013. *Wireframes*. Tillgänglig: http://www.usability.gov/templates/wireframes.pdf Hämtat 21.3.2013

Goltz, Shlomo. 2013. *Creating Wireframes And Prototypes With InDesign.* Tillgänglig: http://www.smashingmagazine.com/2013/03/07/creating-wireframes-and-prototypes-with-indesign/ Hämtat 21.3.2013

Stackoverflow. 2011. *Firefox CSS spacing issues cross-os.* Tillgänglig: http://stackoverflow.com/questions/6022687/firefox-css-spacing-issues-cross-os Hämtad 17.3.2013

Wolski, Marek. 2013. *There Is No Mobile Internet!.* Tillgänglig: http://www.smashingmagazine.com/2013/02/25/there-is-no-mobile-internet/Hämtad 15.3.2013

Cannon, Thomas. 2012. *An Introduction to Color Theory for Web Designers.* Tillgänglig: http://webdesign.tutsplus.com/articles/design-theory/an-introduction-to-color-theory-for-web-designers/ Hämtad 18.3.2013

Google, 2013a. *Contrail One.* Tillgänglig: http://www.google.com/fonts/specimen/Contrail+One#charset Hämtad 18.3.2013

Google, 2013b. *Typography.* Tillgänglig: http://developer.android.com/design/style/typography.html Hämtad 18.3.2013

Google Webfonts. 2013. Tillgänglig: http://www.google.com/webfonts Hämtad: 16.3.2013

HTML5 Boilerplate. 2013. Tillgänglig: http://www.html5boilerplate.com Hämtad: 16.3.2013

Modernizr. 2013. Tillgänglig: http://www.modernizr.com/ Hämtad: 16.3.2013

HeiaHeia. 2013. Tillgänglig: http://www.heiaheia.com Hämtad: 16.3.2013

Sports Tracker. 2013. Tillgänglig: http://www.sports-tracker.com Hämtad: 16.3.2013

Garmin Connect. 2013. Tillgänglig: http://connect.garmin.com Hämtad: 16.3.2013

mScorecard. 2013. Tillgänglig: http://www.mscorecard.com Hämtad: 16.3.2013

Github. 2013. Tillgänglig: http://www.github.com Hämtad: 16.3.2013

Evernote. 2013. Tillgänglig: http://www.evernote.com Hämtad: 16.3.2013

Google+. 2013. Tillgänglig: http://plus.google.com Hämtad: 16.3.2013

Adobe Kuler. 2013. Tillgänglig: http://kuler.adobe.com Hämtad: 16.3.2013

Wikipedia [www]. Hämtat 26.3.2013

Wikipedia, 2013a | https://en.wikipedia.org/wiki/Windows_phone_7

Wikipedia, 2013b | http://en.wikipedia.org/wiki/Sprite_(computer_graphics)

Wikipedia, 2013c | http://en.wikipedia.org/wiki/Website_wireframe

Wikipedia, 2013d | https://en.wikipedia.org/wiki/Responsive_Web_Design

BILAGOR