

Institutionen för datavetenskap  
Department of Computer and Information Science

Examensarbete

**Gymlog, en webbapplikation för personer  
som är vana att träna på gym**

av

**André Engström**

LIU-IDA/LITH-EX-G--13/049--SE

2013-12-17



**Linköpings universitet**

Linköpings universitet  
Institutionen för datavetenskap

Examensarbete

# **Gymlog, en webbapplikation för personer som är vana att träna på gym**

av

**André Engström**

LIU-IDA/LITH-EX-G--13/049--SE

2013-12-17

Handledare: Henrik Eriksson

Examinator: Henrik Eriksson

## Sammanfattning

Syftet med denna rapport är att undersöka om det går att utveckla en användbar webbapplikation som underlättar resultathantering för vana gymutövare. Eftersom rapporten endast fokuserar på vana gymutövare, undkommer webbapplikationen problemet med alldeles för mycket valmöjligheter som gör den osmidig. Vidare undersöks om webbapplikationen även kan motivera användaren till bättre prestationer.

Webbapplikationen är utvecklad med hjälp av de grafiska gränssnittskomponenterna som Smart GWT tillhandahåller. Rapporten innehåller en teoretisk bakgrund till de utvecklingsverktyg som använts för att ta fram webbapplikationen.

I resultatet framgår att det finns potential att ta fram en fullt fungerande webbapplikation för vana gymutövare. I dagsläget går det dock inte att utvärdera webbapplikationen fullt ut då den befinner sig i ett tidigt stadium av utvecklingsprocessen. Då applikationen erbjuder viss inmatning och visualisering av resultat kan detta ge motivation till användaren, genom att de har möjlighet att jämföra sina prestationer. Vidare framgår det i resultatet att det bör finnas ett komplement till inmatning för att det ytterligare ska underlätta resultathanteringen för användaren.

## Förord

*I denna rapport beskrivs mitt examensarbete på högskoleingenjörsutbildningen med inriktning datateknik vid Linköpings tekniska högskola. Jag vill tacka Henrik Eriksson som har tagit sig tid att agerat både handledare och examinator för detta arbete.*

*André Engström*

## Förkortningar och definitioner

**Deferred Binding**, vid kompilering kommer flera versioner av koden att genereras för de vanligaste webbläsarna. Vid access kommer den rätta koden och resurser att köras beroende på vilken webbläsare som används.<sup>1</sup>

**Gränssnittskomponenter**, (eng. widget), ett grafiskt element som illustrerar information som kan ändras av användaren, så som en textruta.<sup>2</sup>

**IDE**, *Integrated development environment* består normalt sett av källkodsredigerare, kompilator och debugger. En IDE tillåter utvecklaren att öka sin produktivitet då den erbjuder att all programmering kan ske i den.<sup>3</sup>

**Java**, objektorienterat programmeringsspråk som vid kompilering skapar bytekod istället för maskinkod, fördelen med detta är att språket kan vara plattformsoberoende. Ursprungligen utvecklat av James Gosling på Sun Microsystems och släpptes 1995.<sup>4</sup>

**JavaScript**, ursprungligen utvecklat som en del av webbläsare så att klientsidans skript skulle kunna interagera med användaren, kontrollera webbläsaren, kommunicera asynkront och modifiera dokumentets innehåll som visades.<sup>5</sup>

**Sandlåda**, (eng. Sandbox), är en säkerhetsmekanism för att separera program som körs. Används ofta för kontroll av otestad och får tillgång till minimalt antal resurser.<sup>6</sup>

**SDK**, "Software Development Kit (SDK) är en uppsättning utvecklingsverktyg som gör det möjligt för mjukvaruutvecklare att bygga applikationer mot ett specifikt programpaket, mjukvaruramverk, hårdvaruplattform, spelkonsol, operativsystem eller liknande".<sup>7</sup>

---

<sup>1</sup> Gwtproject, 2013, *Coding Basics Deferred*

<sup>2</sup> Wikipedia, 2013, *GUI widget*

<sup>3</sup> Wikipedia, 2013, *Integrated development environment*

<sup>4</sup> Schildt och Coward 2012:

<sup>5</sup> Wilton och McPeak 2010:

<sup>6</sup> Wikipedia, 2013, *Sandbox (computer security)*

<sup>7</sup> Wikipedia, 2013, *Software Development Kit*

# Innehållsförteckning

1	Inledning .....	1
1.1	Bakgrund .....	1
1.2	Syfte och frågeställning.....	1
1.3	Avgränsningar .....	1
1.4	Metod .....	1
1.5	Diskussion kring källor .....	2
1.6	Struktur.....	2
2	Teori .....	3
2.1	Google Web Toolkit.....	3
2.1.1	Serverdel.....	3
2.1.2	Klientdel .....	3
2.1.3	Remote Procedure Call.....	3
2.2	Smart GWT .....	5
2.3	Google App Engine .....	6
2.4	Gymträning allmänt.....	7
2.5	Relaterade arbeten .....	8
3	Utförande.....	9
3.1	Utvecklingsmiljö .....	9
3.2	Krav .....	10
3.3	Struktur.....	11
3.4	Grafisk design allmänt .....	12
3.5	Webbapplikationens olika delar .....	13
3.5.1	Användarinloggning .....	13
3.5.2	Ny användare.....	14
3.5.3	Header .....	14
3.5.4	Modifiering av resultat .....	14
3.5.5	Träd-vy .....	16
3.5.6	Visualisering av resultat .....	16
4	Resultat och diskussion .....	17
	Referenser .....	19
	Bilaga 1 Utvärdering gymlog .....	21
	Bilaga 2 Lästips.....	22
	Bilaga 3 exempel på manuell logg .....	23

## Figurförteckning

Figur 1. Visar arkitekturen för RPC.....	4
Figur 2. Portfölj för Smart GWT.....	5
Figur 3. Visar relationen mellan de fem hjälpklasserna.....	11
Figur 4. Visar huvudlayouten för webbapplikationen.....	12
Figur 5. Visar inloggning till webbapplikationen .....	13
Figur 6. Visar headerdelen för webbapplikationen .....	14
Figur 7. Visar tre olika delar för inmatning.....	15
Figur 8. Visar träd-vyn som består av rutiner, träningsupplägg och övningar.....	16
Figur 9. Visar visualiseringen av resultat.....	16

# 1 Inledning

## 1.1 Bakgrund

Det finns idag webbapplikationer som hanterar resultat, men med dessa tillkommer alldeles för mycket valmöjligheter och blir på så sätt osmidiga. Genom dessa valmöjligheter kommer fokus att tas från resultathanteringen. Det kan finnas behov av en webbapplikation som lägger fokus på att underlätta för de som är lite mer vana att träna på gym. Behovet grundar sig i egna erfarenheter men har även framkommit då ämnet diskuterats med andra personer. I denna rapport avser en van gymutövare en person som har tillräckligt med erfarenhet av gymträning för att på egen hand strukturera och utvärdera sina gympass.

## 1.2 Syfte och frågeställning

Med hjälp av denna webbapplikation kommer många att få det enklare att hantera sina resultat på gymmet. De kommer kunna följa sin progression på ett bra och smidigt sätt, vilket troligen kommer motivera dem ytterligare. Då denna webbapplikation lägger fokus på en viss typ av tränande, kommer kanske inte bredden av användare vara stor. Istället för att göra alla hyfsat nöjd anser jag det bättre att göra en viss grupp riktigt nöjd.

**Går det utveckla en webbapplikation som underlättar hanteringen för användarens gymprestationer?**

**Kan webbapplikationen motivera användaren till bättre prestationer?**

**Finns potential för en webbapplikation som är användbar för vana gymutövare?**

## 1.3 Avgränsningar

Till resultat på gymmet kan kroppsmått vara till intresse, men denna del valdes bort för att bevara enkelheten hos webbapplikationen.

För att undersöka om det är möjligt att framställa en webbapplikation för vana gymutövare, framtog endast en prototyp för utvärdering. Prototypen hölls på en grundlig och stilren nivå då fokus inte var i det grafiska gränssnittet. Genom att till denna prototyp fokusera på användarens inmatning och låta visualiseringen vara simpel kan främst inmatningsdelen utvärderas.

## 1.4 Metod

För att få en uppfattning om vad en van gymutövare vill använda sig av i en webbapplikation har utfrågningar gjorts. Utfrågningen har utförts på fem män som har 3-15 års erfarenhet av gymträning. Där har de fått ge idéer på delar som de tycker borde finnas, men även ge återkoppling på olika förslag. Genom dessa idéer har sedan webbapplikationen byggts upp. Genom att komplettera egen kunskap om gymträning och vad andra gymutövare tycker, har en bra grund erhållits för vilka riktlinjer som gäller för webbapplikationen.

Innan utvecklingen startade utfördes en litteraturstudie för att få en bra teoretisk grund. Genom att studera det material som tillhandahölls från utvecklarnas hemsidor kunde utvecklingen påbörjas med hjälp av Eclipse 4.2 och plugin för GWT samt Smart GWT.



Under utvecklingens gång har vissa forum använts som komplement för att lösa problem som uppstått.

För att utvärdera webbapplikationen har personer fått testa att använda applikationen och besvara vissa frågor (se bilaga 1). Eftersom webbapplikationen befinner sig i ett tidig skede av utvecklingsprocessen var det svårt att utvärdera applikationen till fullo. Utvärderingen blev snarare ett vägledning till vad som måste göras för att få den komplett.

## **1.5 Diskussion kring källor**

De källor som har använts är i huvudsak från forum och dokumentation på Googles egna hemsidor gällande GWT och Google App Engine. Projekthemsidan för Smart Gwt innehöll tillräcklig information för att påbörja arbetet. På sidan fanns också rekommenderade länkar att läsa för att få en bättre inblick, däribland den blogg som använts. Den information som har använts tillhörde eller rekommenderades av skaparna, vilket ger källorna hög relevans. Till definitioner har främst Wikipedia använts vilket kan motiveras med att en definition inte kräver lika tunga källor utan dessa ska endast ge läsaren en kort teknisk beskrivning.

På grund av att gymträning som ämne är ett subjektivt område kommer teorin angående detta bestå av egna erfarenheter.

## **1.6 Struktur**

Rapporten består av tre huvudkapitel; *Teori*, *Utförande* samt *Resultat och diskussion*. Kapitlet teori ger läsaren en bakgrund till de utvecklingsverktyg och ramverk som använts i arbetet. Därefter kommer kapitlet *Utförande*, som beskriver systemets uppbyggnad och funktionalitet. Avslutningsvis kommer resultatet samt en diskussion och utvärdering av webbapplikationen.

## 2 Teori

### 2.1 Google Web Toolkit

Google Web Toolkit (GWT) är ett utvecklingsverktyg för utveckling av AJAX-baserade applikationer skriven i Java, som vid kompilering kommer generera JavaScript, HTML och XML-filer. Utvecklaren måste ta hänsyn till att det endast är vissa paket i Java som stöds då koden måste gå att kompilera från Java till JavaScript för att den ska gå att använda. Denna restriktion gäller för kod skriven att köras på klientdelen (se avsnitt 2.1.2).

Den JavaScript som genereras kommer att vara optimerad och effektiv. En stor fördel är att kompilatorn skapar olika JavaScript-filer för de större webbläsarna. Detta låter programmeraren arbeta på en abstraktionsnivå där denne inte behöver ta hänsyn till vilken webbläsare applikationen ska användas. När klienten kör applikationen kommer korrekt JavaScript att laddas ner vilket sker med hjälp av deferred binding (se Förkortningar och definitioner).<sup>8</sup>

Utveckling sker för en serverdel och en klientdel där kommunikation sker med hjälp av ett Remote Procedure Call (RPC).

#### 2.1.1 Serverdel

Kod som körs på serverdelen behöver inte konverteras till JavaScript, vilket medför att utvecklaren kan använda sig av alla paket i Java som denne behöver. Med hjälp av en servlet accepterar servern en förfrågan från en klient, i detta fall en webbläsare som skickar en HTTP request, servern kommer slutligen att generera ett svar i form av HTTP response. Servern används vanligtvis för att lagra och hämta data som användaren behöver.<sup>9</sup>

#### 2.1.2 Klientdel

Här implementeras den del som kommer köras i användarens webbläsare, vilket i stort sett är alla GUI element. Koden kommer vid kompilering att konverteras till JavaScript.

#### 2.1.3 Remote Procedure Call

Remote Procedure Call gör det enkelt för servern och klienten att utbyta objekt. Det skiljer sig till andra traditionella applikationer där hela html-sidor hämtas vid ett anrop från klienten till servern. Istället överförs objekt fram och tillbaka mellan klient och server. På detta sätt har man möjligheten att överföra användargränssnittets logik till klientdelen och på så sätt snabbar man upp applikationen och minskar behovet av bandbredd, men även minimerar belastningen på serverdelen.

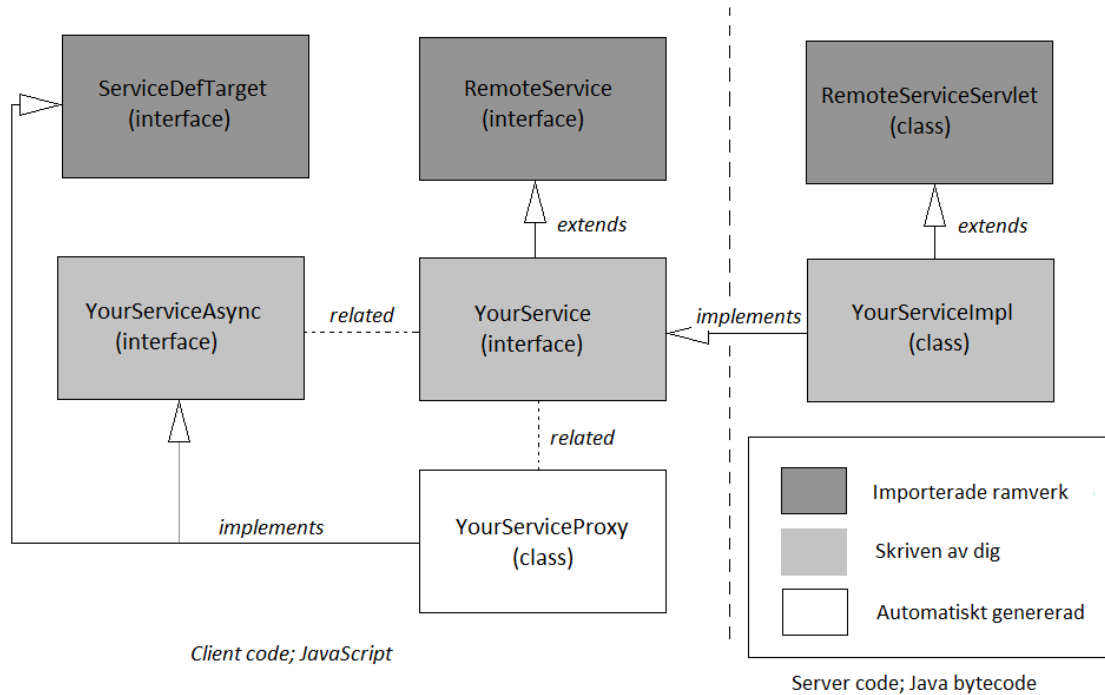
Anropet sker asynkront vilket innebär att klienten inte kommer vänta på att servern kör koden som representerar anropet, utan resultatet kommer via det callback-objektet som skickas som en parameter. När man sätter upp GWT RPC finns det tre element att ta hänsyn till; tjänsten som körs på servern, den kod hos klienten som anropar tjänsten och de Java objekt som ska skickas. Alla objekt som skickas via GWT RPC måste gå att

---

<sup>8</sup> Guermeur, Unruh, och Derrien 2010:

<sup>9</sup> Gwtproject, 2013, *Server Communication*

serialisera, själva serialiseringen sker automatiskt.<sup>10</sup> Proxyn blir automatiskt genererad (se figur 1), "YourServiceAsync" är gränssnittet (eng. interface) som inkluderar de metoder som klienten kan använda för att kommunicera med servern. Även "YourService" är ett gränssnitt men det som skiljer dessa åt är att det förstnämnda även innehåller ett callback-objekt. Detta så att servern kan skicka tillbaka ett svar när den har exekverat den del av koden som motsvarar det anrop klienten gjort.<sup>11</sup>



**Figur 1. Visar arkitekturen för RPC.**

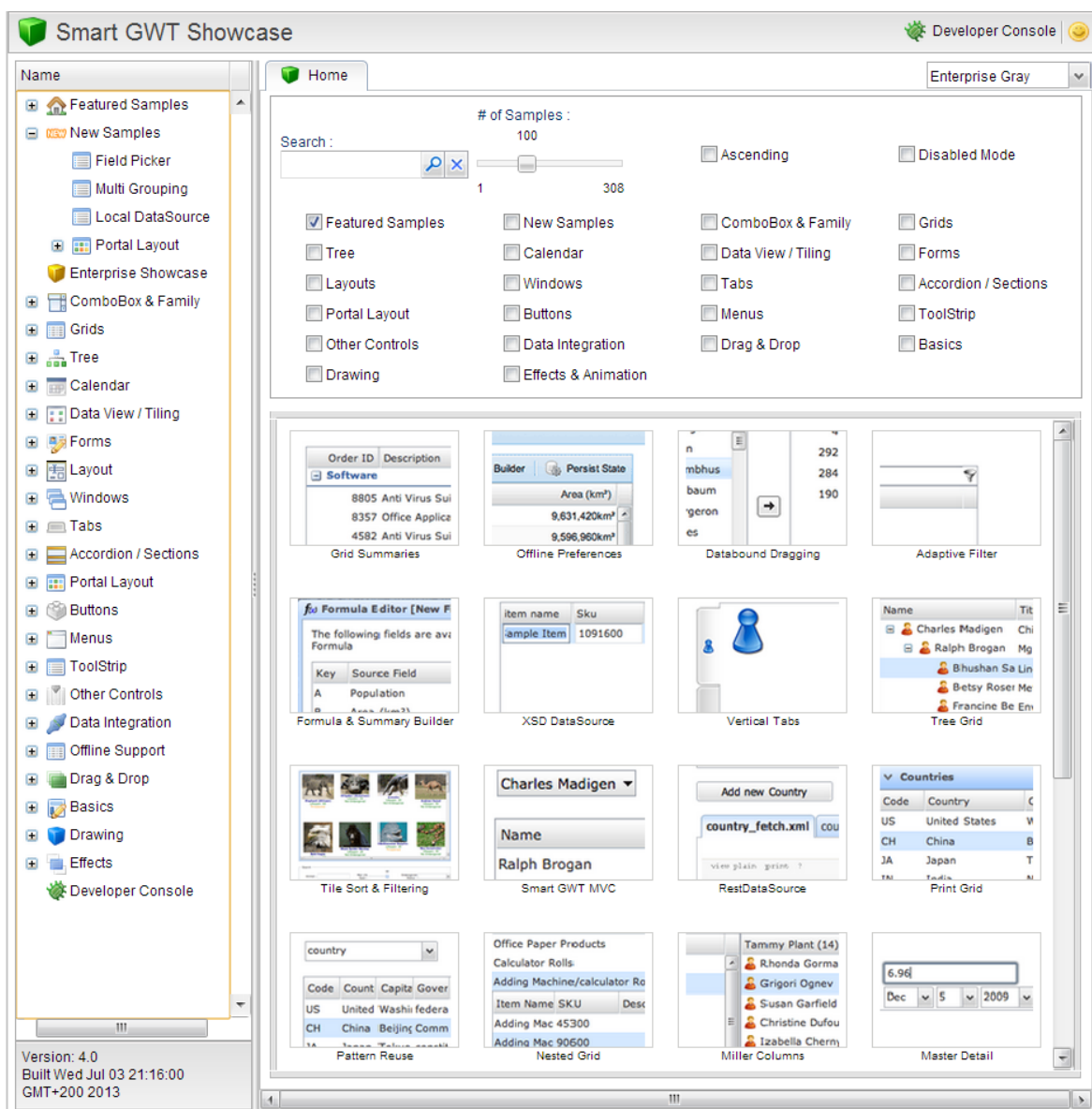
<sup>10</sup> Gwtproject, 2013, *Server Communication*

<sup>11</sup> Google Developers, 2013, *RPC Interfaces*

## 2.2 Smart GWT

Smart GWT är ett GWT baserat ramverk som tillåter utvecklaren att snabbt skapa applikationer med hjälp av biblioteket för gränssnittskomponenter (eng. widget). Komponenterna är designade för att maximera svarstiden men även att minimera serverbelastningen.<sup>12</sup>

En stor fördel med detta ramverk är att den som använder det inte behöver oroa sig över till exempel tillståndshantering eller drag och släpp (eng. drag and drop) eftersom den funktionaliteten redan finns inbyggd i Smart GWT. För att få en överblick vilka gränssnittskomponenter som finns tillgängliga tillhandahålls en portfölj över dessa, som i sin tur är uppbyggd av nära 250 gränssnittskomponenter (se figur 2).<sup>13</sup>



Figur 2. Portfölj för Smart GWT

<sup>12</sup> Isomorphic SOFTWARE, 2012, *Smart GWT Quick Start Guide*

<sup>13</sup> Sjjivan, 2008, *SmartGWT 1.0 Released*

## 2.3 Google App Engine

Google App Engine (GAE) är en plattform och SDK för utveckling och lagring av webbapplikationer på Googles servrar. Valet finns att köra sin applikation helt gratis, då med begränsade resurser, eller betala för sin applikation. Kostnaden är beroende på hur mycket resurser applikationen använder. Genom att specificera en översta gräns för kostnaden kan en eventuell budget hållas, gränsen kan ökas vid behov.<sup>14</sup> Via ett gmail-konto fås tillgång till Google App Engines gratisversion.

Ett gratiskonto kan ha 10 applikationer registrerade och varje applikation kan använda upp till 1 GB lagringsutrymme och får tillgång till tillräckligt med resurser (CPU, bandbredd) för ungefär 5 miljoner sidvisningar per månad (SweClockers.com har ungefär 3 miljoner sidvisningar på en vecka<sup>15</sup>). SweClockers.com är Sveriges största webbtidning om datorer och hårdvara). Då betalvarianten aktiveras höjs dessa gratisnivåer och applikationen börjar kosta först när dessa nivåer överskrids. Nya versioner av en applikation kan publiceras men låta den gamla versionen vara den som används. Då kan den nya versionen testas på GAE, fram till dess att utvecklaren bestämmer sig att köra med den nya versionen istället.

Google App Engine tillåter utveckling i flertalet programmeringsspråk, där Java hör till det vanligaste. Applikationen körs i en sandlåda (se Förkortningar och definitioner) som är en säker miljö med minimerad åtkomst till det underliggande operativsystemet. I och med detta fås en säker och isolerad miljö som är oberoende av placeringen av hårdvaran. Lagringen av data skiljer sig från vanliga relationsdatabaser med att enheter (eng. entities) har en typ och vissa egenskaper. Genom en fråga (eng. query) kan en enheten återfås filtrerad och sorterad efter dess egenskaper. Endast vissa egenskapsvärden tillåts och finns dokumenterade (se bilaga 2).<sup>16</sup> Då index skapas när data ska lagras kommer en uppslagning ta linjär tid med avseende på antalet resultat och inte antalet data som är lagrat. Detta skapar en mycket effektiv läsning även om datalagringen är mycket stor.<sup>17</sup>

---

<sup>14</sup> Guermeur, Unruh, och Derrien 2010:

<sup>15</sup> KIA-index, 2013

<sup>16</sup> Google Developers, 2013, *What Is Google App Engine*

<sup>17</sup> Guermeur, Unruh, och Derrien 2010:

## 2.4 Gymträning allmänt

Gymträning/styrketräning handlar om att lyfta en vikt ett antal gånger genom en muskelkontraktion, vilket framtvingar kroppen att reagera med ökad styrka, storlek och/eller uthållighet.

Terminologi:

- Rep - förkortning för repetition, vilket motsvarar ett lyft av en vikt.
- Set - antalet repetitioner som utförts utan vila mellan.
- Övning - består av antalet set som utförts för en specifik muskelrörelse.
- Träningsupplägg - genom att kombinera övningar fås ett träningsupplägg, vilket utförs under ett träningspass.
- Rutin - består av ett antal träningsupplägg, vilket ofta följer en viss metodik för att uppnå önskat resultat.

För en van gymtövare är ofta målet med ett träningspass att prestera bättre än föregående. Detta genom att förbättra utförandet, öka antalet repetitioner eller lyfta tyngre vikter. Resultat kan memoreras eller noteras i någon form av logg. Då resultat memoreras sker det oftast för någon enskild övning, detta ger då ett riktmärke. Genom att istället föra en logg kan gymtövaren enklare få progression i och med att denna har full koll på hur tidigare träningspass utförts. Det finns olika värden som kan vara intressanta att spara i en träningslogg (se Tabell 1).

En viktig del då en logg förs är att det inte ska ta upp för mycket tid att upprätthålla den. Det ska inte heller finnas allt för mycket extra finesser som tar bort fokus från loggens grundidé, att ge en överblick på progression. Eftersom synen på progression kan variera är det en stor fördel att tillåta sortering av resultaten. Detta kan vara svårare att upprätthålla för de som endast skriver sin logg på papper (se bilaga 3). En teknisk lösning kan i detta fall vara ett bättre alternativ. Progression kan exempelvis innebära att öka antalet repetitioner under ett set, mindre vila mellan seten eller bättre utförande i en övning.

**Tabell 1. Intressanta datatyper för en träningslogg**

Repetitioner	Antal
Belastning	Vikt
Superset	Set från två eller flera olika övningar utförs direkt efter varandra utan vila mellan.
Droppset	Två eller flera set av samma övning, vikten minskas eftersom och utan vila mellan seten.
Tid	Kan innebära tiden mellan set, övning och/eller träningspassets längd.
Kommentarer	Information om träningsutförande, hur träningspasset kändes och/eller noteringar på saker att tänka på till nästa träningspass.

## 2.5 Relaterade arbeten

Det finns ett antal webbapplikationer och webbsidor som tillåter en användare att föra logg över sina träningsresultat.

**Wger, Workout Manager** tillåter användaren att föra en logg över träningsresultat, kost och kroppsvikt. Det finns fördefinierade övningar med tillhörande beskrivning av övningsutförande men för att lägga till egna övningar måste användaren registrera sig. Användaren kan skapa rutiner och lägga till träningspass till dessa, men ges även möjlighet till utskrift av upplägg. Det finns möjlighet att kommentera varje övning med exempelvis hur övningsutförandet kändes eller notera någon förändring som bör göras till nästa träningspass.

**Dumbbell.se** tar det ett steg längre och skapar en sida som inte bara tillåter användaren att föra en logg över sina prestationer utan det finns även diskussionsforum och möjlighet att följa andra användare. I forumet kan användarna diskutera och ge sina åsikter om både kost och träning men även ställa frågor. Ytterligare finns funktionalitet för att lägga upp bilder och filmer.

## 3 Utförande

### 3.1 Utvecklingsmiljö

Eftersom det fanns ett plugin för GWT till eclipse blev det ett givet val av IDE för utveckling av webbapplikationen. Med detta plugin blir det enkelt att skapa nya projekt med autogenererade filer som behövs för att snabbt starta utvecklingen. Det tillåter även att man publicerar applikationer på GAE genom det medföljande uppladdningsverktyget.

Eclipse 4.2 (Juno) kördes på en stationär dator med Linux Ubuntu 12.04 (lts).

För motverka förlust vid en eventuell trasig hårddisk eller liknande använde jag mig av Ubuntu One som är en gratis molntjänst för lagring och synkronisering av data. Via gratisversionen ges 5 GB utrymme som kan utökas om man väljer att betala för tjänsten. Ubuntu One är enkel att ställa in genom välja de filer som ska lagras på molnet vilka synkroniseras vid förändring.



## 3.2 Krav

Kraven för webbapplikationen är uppdelade i tre nivåer, där nivå 1 är krav med högst prioritet och nivå 3 har lägst prioritet. Graderingen av kraven återspeglas i vad som är rimligt att åstadkomma inom tidsramen för detta arbete och för att kunna dra slutsatser och diskutera kring frågeställningarna.

### Nivå 1

- Inloggning av befintlig användare
- Nyskapande av användare
- Resultat på servern och klienten ska vara synkroniserade
- Gränssnittskomponenter ska hålla sig uppdaterade då resultat ändras
- Användaren ska kunna lagra information om rutiner, träningsupplägg, övningar och set
- Skapa och ta bort rutiner, träningsupplägg och övningar
- Ett set ska bestå av en vikt och antalet repetitioner vikten lyfts

### Nivå 2

- Visualisering av resultat i en graf
- Sortera resultat i tabellen
- Header-del som visar viss information för användaren
- Möjlighet att visa och dölja hjälpinformation för hur webbapplikationen fungerar
- Modifiera befintliga set

### Nivå 3

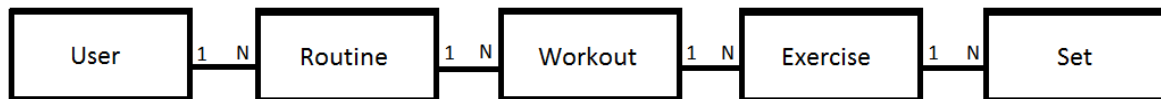
- Utskrift av träningsupplägg
- Kalender
- Inloggning med OpenID

Kraven beskrivna i nivå 3 har inte implementerats för detta arbete då det ej funnits tid och har därmed inte prioriterats. Alla krav i nivå 1 och 2 har implementerats, med undantag för "Visualisering av resultat i en graf" under nivå 2. Genom att resultat redan visas i en tabell har funktionaliteten för att visualisera resultat i graf ej implementerats.

### 3.3 Struktur

Webbapplikationen innehåller fem stycken hjälpklasser (se figur 3) för kontroll av användarresultat. Hela webbapplikationen använder sig av dessa klasser för att visualisera och modifiera resultat. När en användare har loggat in kommer den översta klassen (User) att sättas som aktiv användare och via den kan webbapplikationen få ut all information om användaren, inklusive resultat.

1. User - innehåller information om användaren och lagrar alla rutiner (Routine) i en array. I klassen finns det metoder för att hämta och ändra värden. Klassen kommer att hämtas och lagras i databasen på serversidan och genom att alla rutiner lagras i klassen kan användarens alla värden uppdateras och användas genom denna. Användarnamnet är unikt och är primärnyckel för hämtning och lagring av klassen.
2. Routine - består av namnet på rutinen och innehåller en array med träningsupplägg (Workout). Det kan endast finnas en rutin med samma namn för en användare. Namnet på rutinen kommer fungera som primärnyckel då en viss rutin vill användas.
3. Workout - liknar Routine men skiljer sig i att arrayen istället består av övningar (Exercise).
4. Exercise - uppbyggd på samma sätt som Routine och Exercise, men arrayen innehåller set (Set).
5. Set - Ett set består av en vikt och antalet repetitioner som utförs. Då ett set generellt inte har ett namn har en primärnyckel fått auto-genereras med hjälp av inbyggd generator.

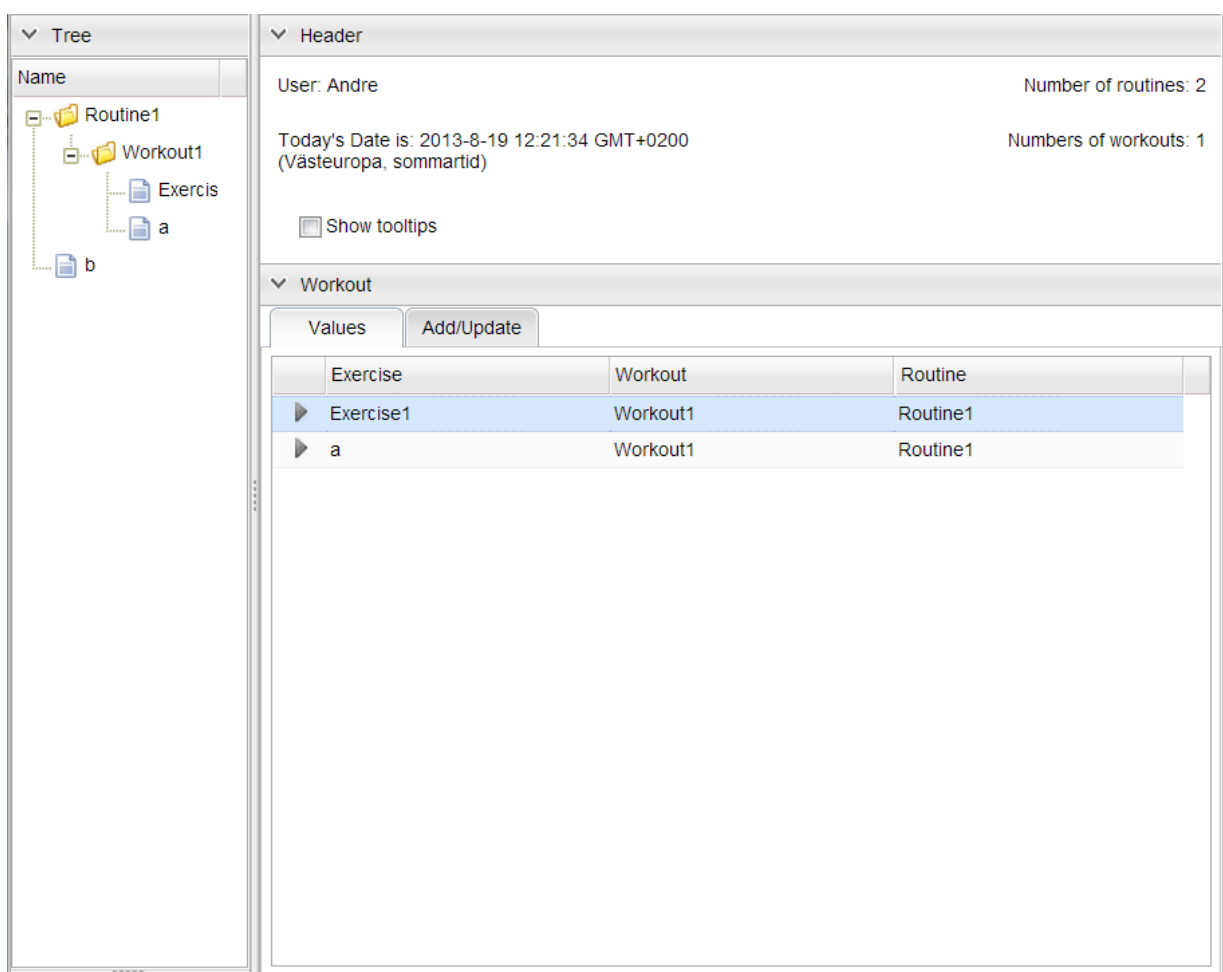


Figur 3. Visar relationen mellan de fem hjälpklasserna

### 3.4 Grafisk design allmänt

Eftersom fokus i detta projekt var användbarheten och inte utseende på webbapplikationen har det grafiska gränssnittet hållits på en relativt grundlig nivå. De grafiska komponenterna visualiseras med den standard som Smart GWT erhåller.

Genom att den grafiska layouten togs fram först kunde allt fokus istället läggas över på funktionaliteten. Huvudlayouten för webbapplikationen delades in i tre delar; Tree, Header och Workout (se figur 4). Workout-delen är den del som hanterar användarens värden genom att antingen visualisera eller uppdatera dem. För att visualisera värden väljer användaren fliken Values och uppdatering sker under fliken Add/Update. Användaren kan välja att minimera eller maximera de tre olika delarna då dessa är uppbyggda av en SectionStack.



Figur 4. Visar huvudlayouten för webbapplikationen

## 3.5 Webbapplikationens olika delar

Genom att kombinera GWT (se avsnitt 2.1), Smart GWT (se avsnitt 2.2) och GAE (se avsnitt 2.3) har webbapplikationen utvecklats.

### 3.5.1 Användarinloggning

Inloggningen till webbapplikationen är väldigt simpel till utseendet, vilket också fungerar som startsida (se figur 5). För att lagra och komma åt sina resultat måste användaren logga in på sitt användarkonto. När lösenordet skrivs in kommer inte tecknen att visas utan dessa kommer att döljas och istället visas som små runda cirklar. Om konto saknas kan detta skapas genom att trycka på knappen "Create new", vilket tar användaren till delen som hanterar nyskapande av användare.

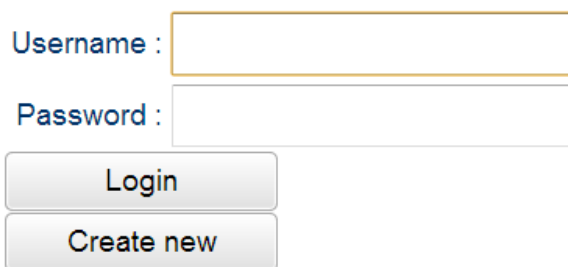
När användaren har angett sitt användarnamn, lösenord och slutligen tryckt på "Login" knappen kommer inloggningsuppgifterna att verifieras. För att verifiera inloggningsuppgifterna görs ett Remote Procedure Call då verifieringen måste ske på serversidan.

Ett asynkront callback-objekt skapas;

```
AsyncCallback<User> callback = new AsyncCallback<User>() {...}
```

Till detta callback-objekt överskuggas två funktioner; onSuccess() och onFailure(). Det förstnämnda kommer att anropas om den metod som körs på servern lyckas returnera. Den andra kan användas för att fånga undantag som utvecklaren valt att kasta vid eventuella problem på serversidan. Detta skickas tillsammans med användarnamn och lösenord till motsvarande metod på servern som hanterar inloggning.

Servern kontrollerar om en objekt User finns lagrad med användarnamnet som primärnyckel. Vid två fall av tre kommer servern att returnera null, vilket motsvarar att inloggningen misslyckades. Detta sker antingen då användaren har angett fel lösenord eller om användaren ej existerar. Det tredje fallet motsvarar att inloggningen lyckades och servern kommer returnera User till klienten. I funktionen onSuccess() görs kontroll om värdet är null eller om inloggningen lyckades. Då värdet är null kommer en dialogruta visas för användaren att inloggningsuppgifterna är felaktiga. När inloggning lyckas kommer användaren att få tillgång till huvuddelen av applikationen och har då tillgång till sin träningsdata. Detta sker till skillnad från mer originell webbprogrammering då en ny html-sida laddas. Istället kommer inloggningen döljas och huvuddelen visas, detta genom att användargränssnittet redan är lagrat hos klienten.



Username :

Password :

Login

Create new

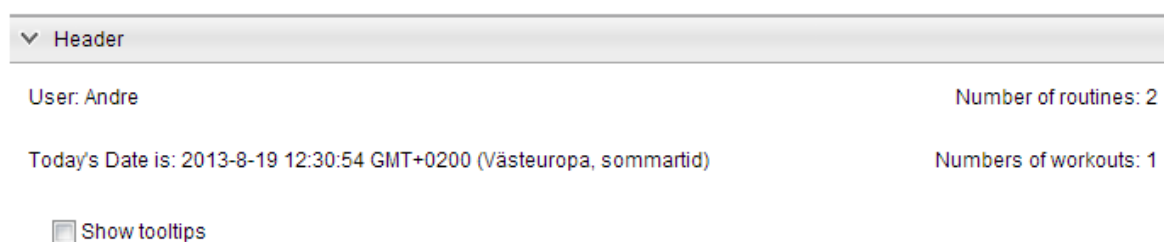
Figur 5. Visar inloggning till webbapplikationen

### 3.5.2 Ny användare

Ny användare skapas på liknande sätt som när en användare loggat in. Skillnaden är vilken metod som kommer anropas på servern. Metoden kontrollerar att användarnamnet inte redan finns registrerat innan den lägger till den nya användaren. Här kommer null att returneras om användaren redan existerar och klassen User om användaren kunde skapas.

### 3.5.3 Header

Genom att använda en header-del kan intressanta värden för användaren visas (se figur 6). När användaren har loggat in visas användarinformationen; användarnamn, antalet rutiner och antalet träningspass. Utöver visas även dagens datum inklusive den lokala tiden vilket uppdateras med ett intervall om 10 sekunder. För att ge användaren en snabb intro till hur applikationen fungerar finns möjligheten att kryssa i "Show tooltips" vilket kommer skriva ut hjälptext för olika delar.



Figur 6. Visar headerdelen för webbapplikationen

### 3.5.4 Modifiering av resultat

Delen som hanterar inmatning är uppbyggd i tre sektioner (se figur 7). I den första delen läggs resultat till genom att trycka på knappen "New" vilket skapar en ny ruta för inmatning av en sträng som representerar namnet på en ny rutin, träningsupplägg eller övning. För att lägga till ett nytt träningsupplägg måste användaren först välja till vilken rutin som träningsupplägget tillhör, detta genom att antingen skapa en ny rutin eller välja bland de befintliga rutinerna. Genom att alla rutiner kommer att vara tillgängliga i rutinboxen kommer användaren snabbt kunna välja till vilket rutin träningsupplägget tillhör. På samma sätt kommer en övning att läggas till, först genom att specificera en rutin följt av ett träningsupplägg för att sedan trycka på tillhörande "New-knapp". De värden som läggs till i denna del kommer direkt att lagras för användaren i databasen.

Genom att välja rutin, träningsupplägg eller övning, följt av "Delete-knappen" kommer tillhörande del tas bort för användaren. Då till exempel en rutin tas bort kommer rutinens alla träningsupplägg och övningar också tas bort.

Den andra delen består av en tabell som hanterar inmatning av set. Genom att använda en tabell kommer användaren få en bra överblick på set tillhörande den valda övningen. Beroende på om den valda övningen är ny eller inte kommer tabellen att innehålla eventuellt befintliga set; dessa kan här modifieras. Ett nytt set läggs till genom att trycka på knappen "New set", vilket kommer skapa en nya rad i tabellen för inmatning. För att ta bort ett set används minusknappen i högra delen av tabellen. Till skillnad från den första delen kommer värden inte att läggas till direkt till databasen. Genom att trycka på knappen "Add exercise" kommer resultaten överförs till en överblickstabell som är till för att användaren ska få en bra överblick över de tänkta förändringar.

Den tredje och sista delen består av överblickstabellen som visar alla förändringar som användaren gjort, men som inte är överfört till databasen ännu. Dessa värden uppdateras i databasen genom att trycka på knappen "Add workout". Genom att inte uppdatera värden flytande utan istället göra det när användaren gjort klart avsedda förändringar; blir antalet accesser till servern betydligt mindre vilket minskar belastning. Ytterligare en fördel är att användaren enkelt kan ha överblick på sina förändringar och snabbt kunna modifiera dessa då de visualiseras i en tabell.

Workout

ValuesAdd/Update

1

Routine : Routine1NewDelete

Workout : Workout1NewDelete

Exercise : Exercise1NewDelete

Repetitions(#)	Weight(kg)	
10	50	⊖
545	10	⊖
33	4	⊖

2

New setAdd exerciseAdd Workout

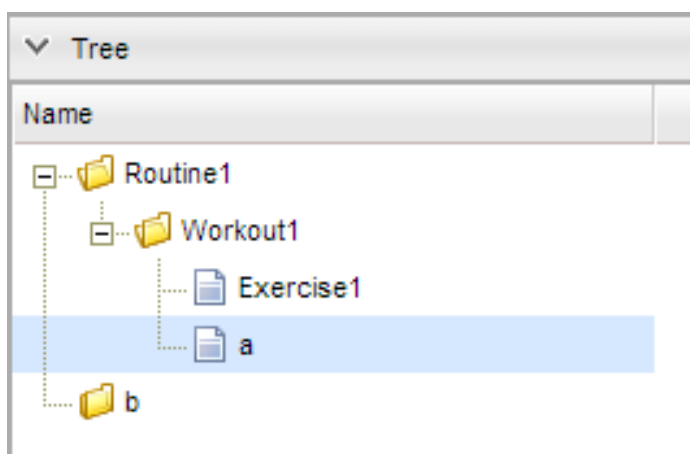
Exercise	Workout	Routine	
Exercise1	Workout1	Routine1	⊖
10 reps @ 50kg			
545 reps @ 10kg			
33 reps @ 4kg			

3

Figur 7. Visar tre olika delar för inmatning

### 3.5.5 Träd-vy

Genom att träd-vyn visualiserar befintliga rutiner, träningsupplägg och övningar kan användaren få en bra överblick för sina träningsresultat (se figur 8). Denna kommer att uppdateras i takt med att användaren uppdaterar sina värden (se avsnitt 3.3.4). I och med att träd-vyn håller sig uppdaterad kommer användaren få en bra överblick på sina resultat oberoende på var i applikationen denne befinner sig. Om användaren är i vyn som hanterar modifiering av resultat kommer den rutin, träningsupplägg och övning som markeras i träd-vyn att sättas som värden i delen som hanterar inmatning. I och med översikten som träd-vyn tillhandahåller får användaren möjlighet att snabbt modifiera sina resultat.



Figur 8. Visar träd-vyn som består av rutiner, träningsupplägg och övningar

### 3.5.6 Visualisering av resultat

För att visualisera resultat för användaren används en tabell (se figur 9). Resultaten kan sorteras efter rutin, träningsupplägg eller övning; på så sätt kan användaren jämföra sina resultat baserat på valet av sortering. De resultat som användaren har, det vill säga set i en övning, kan visas genom att expandera raden i tabellen som övningen finns. Där kommer alla set som övningen består av att visas med formatet:

*N reps @ M kg* (N - antalet repetitioner, M - vikt).

The image shows a table interface with a 'Workout' header and a 'Values' button. The table has three main columns: 'Exercise', 'Workout', and 'Routine'. The first row is expanded, showing details for 'Exercise1'. The details are listed as '10 reps @ 50kg' and '545 reps @ 10kg'. The second row shows 'a' under the 'Exercise' column, 'Workout1' under the 'Workout' column, and 'Routine1' under the 'Routine' column.

Exercise	Workout	Routine
Exercise1 10 reps @ 50kg 545 reps @ 10kg	Workout1	Routine1
a	Workout1	Routine1

Figur 9. Visar visualiseringen av resultat

## 4 Resultat och diskussion

Den framtagna webbapplikationen är i dagsläget inte fullt användbar då den är i ett tidigt skede av utvecklingsprocessen. Eftersom det endast funnits tid att ta fram en prototyp som testats men inte vidareutvecklats, saknar den mycket funktionalitet som är nödvändig för att den ska fungera bra för användaren. Jag anser dock att den har god potential att vidareutvecklas till en fullt fungerande webbapplikation för vana gymutövare. Med mer tid kan applikationen vidareutvecklas och användartestas ytterligare för att på så sätt kunna ta fram en fullt fungerande version.

Vad som framgår från efterstudien (se bilaga 1) är att testpersonerna tycker att webbapplikationen har potential men saknar en del funktionalitet för att den ska vara användbar. Ytterligare tid bör läggas på det grafiska gränssnittet för att den ska vara lite mer tilltalande. Vidare efterfrågas en graf för att enklare kunna se progressionen då den befintliga tabellen inte riktigt ger bästa överblicken. Själva inmatningen känns bra men går att förbättra ytterligare, främst genom att alltid ha tillgång till alla tillagda övningar i systemet och inte bara de som tillhör det angivna träningsupplägget. Då kan exempelvis övningar som redan tillhör träningsupplägget markeras.

### **Går det utveckla en webbapplikation som underlättar hanteringen för användarens gymprestationer?**

Webbapplikationen saknar delar som bör finnas för att användaren ska få det enklare att hantera sina gymprestationer. Den del som nu är mest utvecklad är inmatningen av resultaten men då resultaten endast visas i en tabell är det svårt för användaren att få en bra överblick över sina prestationer. Delen som hanterar inmatning ger användaren en mycket flexibel och snabb lagring av sina resultat. Vidare bör någon ytterligare visualiseringsform av data erbjudas. Ett bra alternativ här skulle kunna vara grafer så att användaren kan få bättre överblick över sina resultat. Här kan även logik för att sortera efter prestationer; som till exempel maxlyft vara av intresse för användaren.

### **Kan webbapplikationen motivera användaren till bättre prestationer?**

Genom att användaren får en överblick över sina prestationer och kan jämföra resultat från olika träningspass kan det ge motivation för att prestera bättre. För att verkligen ge motivation till användaren måste möjligheten att visualisera resultaten förbättras. Det måste finnas bättre möjlighet att göra jämförelser mellan resultat med avseende på olika kriterier.

### **Finns potential för en webbapplikation som är användbar för vana gymutövare?**

I det stadium som webbapplikationen är idag är den inte användbar för en van gymutövare då den saknar mycket funktionalitet som bör finnas. Men det finns stor potential att vidareutveckla en sådan webbapplikation med den framtagna som bas. Den version som finns nu gör det möjligt för användaren att lagra resultat genom att inmatningen fungerar och resultaten kommer att lagras för användaren i databasen. På så sätt får användaren en möjlighet till att i alla fall lagra resultaten men även ha möjligheten att se dem på ett överskådligt sätt, dock inte optimalt.



Om en jämförelse görs med de relaterade arbeten som beskrivs i rapporten, (se avsnitt 2.5), där jag anser att det finns lite för mycket extra funktioner som sätter fokus på annat än progression i gym prestationerna. Dessa extra funktioner, så som övningsutföranden och forum, kan underlätta för personer som inte är van att träna på gym. Dessa medför ingen större hjälp för en person som är van att träna på gym, utan kommer snarare att vara överflödig och i värsta fall skapa ett irritationsmoment för användaren. Då webbapplikationen inte innehåller information om hur du tränar eller vad du ska träna tror jag den ämnar sig bättre för just personer som är van att träna på gym.

Jag anser att det bör finnas ett komplement till webbapplikationen i form av en mobilapplikation. Med hjälp av den kan då användaren direkt föra in sina resultat på gymmet vilket då kan synkroniseras med webbapplikationen. Då kan användaren på ett mycket enkelt sätt föra in sina resultat och har då möjlighet att via webbapplikationen få bra översikt på sina resultat.

## Referenser

- Guermeur, Daniel, Amy Unruh, och Dom Derrien. 2010. *Google app engine java and GWT application development [elektronisk resurs] : Build powerful, scalable, and interactive web applications in the cloud / daniel guermeur, amy unruh ; reviewers, dom derrien ... [et al.]*Birmingham, U.K. : Packt Pub., 2010 (Norwood, Mass. : Books24x7.com generator)].
- Schildt, Herbert, och Danny Coward. 2012. *Java [elektronisk resurs] : A beginner's guide, fifth edition / herbert schildt ; technical editor, danny coward*New York : McGraw-Hill, c2012 (Norwood, Mass. : Books24x7.com generator)]; 5th ed.
- Wilton, Paul, och Jeremy McPeak. 2010. *Beginning JavaScript [elektronisk resurs] / paul wilton, jeremy McPeak*Hoboken, NJ : Wiley, c2010; 4th ed.
- Google Developers (2013). *RPC Interfaces*.  
< [https://developers.google.com/eclipse/docs/gwt\\_rpc](https://developers.google.com/eclipse/docs/gwt_rpc)>  
Hämtad 2013-07-10
- Gwtproject. (2013). *Coding Basics Deferred*.  
<<http://www.gwtproject.org/doc/latest/DevGuideCodingBasicsDeferred.html>>.  
Hämtad 2013-07-10
- Google Developers. (2013) *What Is Google App Engine*.  
< <https://developers.google.com/appengine/docs/whatisgoogleappengine>>  
Hämtad 2013-07-16
- Gwtproject. (2013). *Server Communication*.  
<<http://www.gwtproject.org/doc/latest/DevGuideServerCommunication.html#DevGuideGettingUsedToAsyncCalls>>.  
Hämtad 2013-07-10
- Isomorphic SOFTWARE. (2012). *Smart GWT Quick Start Guide*.  
< [http://www.smartclient.com/releases/SmartGWT\\_Quick\\_Start\\_Guide.pdf](http://www.smartclient.com/releases/SmartGWT_Quick_Start_Guide.pdf)>  
Hämtad 2013-07-11
- Kian-index. (2103).  
<<http://kiaindex.net/l/yw:201326/c:15>>  
Hämtad 2013-07-16
- Sjivan. (2008). *SmartGWT 1.0 Released*.  
< [http://www.jroller.com/sjivan/entry/smartgwt\\_1\\_0\\_released](http://www.jroller.com/sjivan/entry/smartgwt_1_0_released)>  
Hämtad 2013-07-11
- Wikipedia. (2013). *GUI widget*.  
< [http://en.wikipedia.org/wiki/GUI\\_widget](http://en.wikipedia.org/wiki/GUI_widget)>  
Hämtad 2013-07-11
- Wikipedia. (2013). *Integrated development environment*.  
<[http://en.wikipedia.org/wiki/Integrated\\_development\\_environment](http://en.wikipedia.org/wiki/Integrated_development_environment)>  
Hämtad 2013-07-16

Wikipedia. (2013). *Sandbox (computer security)*.  
< [http://en.wikipedia.org/wiki/Sandbox\\_\(computer\\_security\)](http://en.wikipedia.org/wiki/Sandbox_(computer_security)) >  
Hämtad 2013-07-16

Wikipedia. (2013). *Software Development Kit*.  
< [http://sv.wikipedia.org/wiki/Software\\_Development\\_Kit](http://sv.wikipedia.org/wiki/Software_Development_Kit) >  
Hämtad 2013-07-16

## Bilaga 1 Utvärdering gymlog

Träningsvana (antal år) \_\_\_\_\_

Träningsfrekvens (dagar/vecka) \_\_\_\_\_

1. Första intryck, webbapplikationen.

2. Hur känns inmatning/uppdatering av resultat?

3. Kan webbapplikationen hjälpa dig som gymutövare?

Förslag på förbättringar:

## Bilaga 2 Lästips

Google Developers. (2013) *Entities, Properties and Keys*

<[https://developers.google.com/appengine/docs/java/datastore/entities#Java\\_Properties\\_and\\_value\\_types](https://developers.google.com/appengine/docs/java/datastore/entities#Java_Properties_and_value_types)>

**Under rubriken "Properties and value types" finns de värden på entities som stöjds för lagring i google app engine.**

## Bilaga 3 exempel på manuell logg

Axlar	s	28/4 -11	6/5 -11	15/6 -11	21/6 -11	27/5 -11
Avelpress smith	1	55x15	60x15	65x15	60x15	67,5x15
	2	65x15	65x15	70x15	60x14	72,5x15
	3	75x12	70x15	75x15	60x10	77,5x15
	4	85x8	75x10	80x8	50x13	82,5x10
	5	55x15	60x15	65x15		
Shrugs hantlar	1	41x12	31x12	33x12	36x12	39x12
	2	41x10	33x12	36x12	39x12	41x12
	3	39x12	36x10	39x12	41x12	43x11
(fr)						
Frammät- lyft viktplatta	1	15x12	15x12	15x12	15x12	15x12
	2	15x10	15x12	15x12	15x12	15x12
	3	15x12	15x12	15x12	15x12	15x12
Sidelöft hantlar	1	11x15	11x15	11x15	11x15	11x15
	2	11x12	11x12	11x13	11x14	11x12
	3	11x10	8x15	8x15	8x15	8x12
Hanteldrag Kabel fäst på bänk	1	15x12	15x10	15x15	17,5x14	17,5x15
	2	12,5x12	12,5x15	15x15	15x15	17,5x15
	3	10x15	12,5x15	15x14	15x15	17,5x15
Rotaförkuff mot bänk	1	4x15	5x15	6x15	7x15	8x15
	2	5x15	6x15	6x15	7x15	8x15
	3	6x15	6x15	6x15	7x15	8x15

## På svenska

Detta dokument hålls tillgängligt på Internet – eller dess framtida ersättare – under en längre tid från publiceringsdatum under förutsättning att inga extra-ordinära omständigheter uppstår.

Tillgång till dokumentet innebär tillstånd för var och en att läsa, ladda ner, skriva ut enstaka kopior för enskilt bruk och att använda det oförändrat för ickekommersiell forskning och för undervisning. Överföring av upphovsrätten vid en senare tidpunkt kan inte upphäva detta tillstånd. All annan användning av dokumentet kräver upphovsmannens medgivande. För att garantera äktheten, säkerheten och tillgängligheten finns det lösningar av teknisk och administrativ art.

Upphovsmannens ideella rätt innefattar rätt att bli nämnd som upphovsman i den omfattning som god sed kräver vid användning av dokumentet på ovan beskrivna sätt samt skydd mot att dokumentet ändras eller presenteras i sådan form eller i sådant sammanhang som är kränkande för upphovsmannens litterära eller konstnärliga anseende eller egenart.

För ytterligare information om Linköping University Electronic Press se förlagets hemsida <http://www.ep.liu.se/>

## In English

The publishers will keep this document online on the Internet - or its possible replacement - for a considerable time from the date of publication barring exceptional circumstances.

The online availability of the document implies a permanent permission for anyone to read, to download, to print out single copies for your own use and to use it unchanged for any non-commercial research and educational purpose. Subsequent transfers of copyright cannot revoke this permission. All other uses of the document are conditional on the consent of the copyright owner. The publisher has taken technical and administrative measures to assure authenticity, security and accessibility.

According to intellectual property law the author has the right to be mentioned when his/her work is accessed as described above and to be protected against infringement.

For additional information about the Linköping University Electronic Press and its procedures for publication and for assurance of document integrity, please refer to its WWW home page: <http://www.ep.liu.se/>