En rundtur med Google Glass

Examensarbete inom Högskoleingenjörsprogrammet i datateknik

Marcus Thorström
Oskar Selberg

Institutionen för data- och informationsteknik
CHALMERS TEKNISKA HÖGSKOLA
Göteborg, Sweden 2015

En rundtur med Google Glass

Marcus Thorström
Oskar Selberg

Department of Computer Science
Chalmers University of Technology
SE-412 96 Göteborg
Sweden
Telephone + 46 (0)31 - 772 10 00

Omslag:
Tim.Reckmann, 2014
http://commons.wikimedia.org/wiki/File:Google_Glass_Main.jpg?uselang=sv

Abstract

Intuitive office environments that make life easier for employees at the same time as the company is saving money, is something that consultant firms have begun showing interest in. The only limit for office design is the price. Sigma IT/Management in Gothenburg gladly shows their office to customers and corporation partners, but have noticed that much of the employees time has to be reserved for the guided tours, and the tours disturbs the employees that are working. This is why the company want to explore the possibility to automatize the tour with the newest smart product on the market, Google Glass.

This project includes the development of a system that enable the company to create, maintain and carry out the office tour, with the help of smart glasses and a website. On the website a user can create and maintain the tour. The glasses purpose is to substitute a human guide.

Tours will be carried out, after the system development, with both smart glasses and a human guide. This so that the two different methods can be evaluated and compared.

The smart glasses will show that the company is at the cutting edge of development, and demonstrate the possibilities of this new product. This in turn will help free employees time, and with the help of the built in speaker not disturb the surroundings.

The result from the project is that a system with a database, website, and Google Glass application was created. To be able to determine what is needed from the application, without prior experience, a user survey was carried out. To develop a system that is meant to replace a human has shown to have drawbacks, such as less input of information, and more time spent on developing the systems layout. This report also includes a discussion about improvements that can be made to reduce disadvantages and increase the benefits.

Keywords: Google Glass, Guided tour, Application, QR-Code, Database, Web Application, First Time User.

Sammanfattning

Intuitiva kontor som underlättar för anställda samtidigt som företag sparar pengar är något som stora konsultföretag nu har börjat intressera sig för. Det enda som sätter gränser för hur kontor kan utformas är kostnaden. Sigma IT/Management i Göteborg vill gärna visa upp sina kontor för kunder och samarbetspartners, men märker att mycket av de anställdas tid behöver avsättas för uppvisning av kontorslandskapet och att dessa uppvisningar stör de anställda som arbetar. Därför vill företaget utforska möjligheten att automatisera rundvandringen med hjälp av den nyaste smarta produkten på marknaden, Google Glass.

Detta arbete inkluderar utveckling av ett system som möjliggör för användare att skapa, underhålla och genomföra rundvandringar med hjälp av smarta glasögon och en hemsida. På hemsidan kan en användare skapa och underhålla rundturen. Glasögonens syfte är att ersätta en mänsklig guide för att rundturen ska bli automatiserad.

Efter systemutvecklingen kommer rundturer med besökare att hållas dels med glasögon och dels med mänsklig guide för att en jämförelse ska göras mellan de två olika metoderna.

De smarta glasögonen visar att företaget ligger i framkant för utveckling, samt att de möjligheter som glasögonen innebär kan visas upp. Glasögonen kommer att frigöra anställdas tid och med inbyggda högtalare bidra till att omgivningen inte störs.

Resultatet av projektet är att det skapats ett system med databas, hemsida och applikation till Google Glass. För att klarlägga vad som krävs av systemet för att en förstagångsanvändare ska klara av att använda det, utan tidigare erfarenhet, har en användarundersökning genomförts. Att utveckla ett system som ersätter en människa har visat sig ha nackdelar, så som minskat informationsintag. Rapporten inkluderar även förslag till förbättringar som kan göras för att reducera nackdelarna och öka fördelarna.

Nyckelord: Google Glass, Guidad tur, Applikation, QR-kod, Databas, Webb Applikation, Förstagångsanvändare.

Författarnas tack

Rapportförfattarna vill tacka projekthandledaren Håkan Burden för hans konsultation under projektets gång. Stort tack till Joakim Jonsson och Rashwan Lazkani för den tekniska hjälp de bistod med. Ytterligare ett tack riktas till Linus Vidberg på Sigma IT/ Management i Göteborg som utlyste och planerade examensarbetet.

Rapportförfattarna vill även tacka Per-Erik Nilsson och My Huttunen för deras hjälp med korrekturläsningen av rapporten, samt alla de personer som besökte Sigma för att genomföra en rundvandring.

Innehållsförteckning

Ordlista

.pdf Filformat för PDF-filer.

Agilt Arbetsmetod för nära samarbete med kunden vilket ger korta iterativa förbättringar.

Async Task Uppgift som sker i bakgrunden på Android.

Backlog Samling uppgifter som skall göras i projektet för att det ska vara helt färdigställt.

Bugs Fel i koden som uppmärksammats och behöver lösas.

Checked in Kod som finns tillgänglig för alla att testa på versionshanteringsystemet.

Clean Desk Policy Anställda har ej fast skrivbordsplacering utan väljer ett nytt varje dag.

Code review Kategori där en systematisk genomgång av källkod krävs.

ContentType Innehållets typ, ev. filformat.

Developer team Personer som sköter utveckling av mjukvara.

Development Utveckling och implementering av funktioner.

Done När en *Scrum master* eller *product owner* är nöjd med en funktion.

Done&done När både *Scrum master* och *product owner* är nöjd med en funktion.

Encoding Kodning av en speciell fil.

Google Glass Huvudmonterad dator utvecklad av Google.

HTTP-Get Metod för att via *Hyper Text Transfer Protocol* göra förfrågningar till en webb-service.

Key Nyckel för identifiering av enskilda rader i en databas.

Open Source Öppen källkod, ej copy-wright skyddat, tillåtet att kopiera och modifiera.

Open workspace Arbetsplats där det inte finns några väggar, på så sätt ser kontoret mer öppen ut.

Otypat Språk som ej kräver fasta typer för variabler utan accepterar vilken typ som helst.

Product owner	Personen som äger slutprodukten och kan därför ha synpunkter på funktioner och design som implementeras i projektet.
Scrum	Utvecklingsmetod där man arbetar iterativt med att ta fram mjukvara.
Scrum master	Person som bestämmer över vad som skall ingå i varje sprint och även ser till att arbetet blir korrekt utfört.
Scrumboard	Visuell representation av en sprint, olika uppgifter är representerade i det stadie de befinner sig.
Sprint	Ett intervall under utvecklingsfasen där inga ändringar får presenteras och det arbete som är planerat slutförts.
Sprint planning	Planering av nästkommande sprint.
Sprint review	Presentation av den senaste sprinten, informellt möte involverande alla i Scrumprojektet.
Task	En uppgift på en Scrumboard.
Testing	Systemets olika funktioner testas på dess funktionalitet.
Trello	Internetbaserat verktyg för att underlätta arbeta med utvecklingsmetoden Scrum.
User stories	Användarberättelser, hur användare skall uppfatta systemet när det är färdigt.
Waterfall-metoden	Metod för utveckling där allt görs steg för steg istället för Agilt.
Weak-entity	En post i en databas som kräver ett annat objekt för att kunna enskilt identifieras.
Webbplats	Samling sammanhängande dokument, texter och bilder som är tillgänglig på Internet.

1 Introduktion

Det kommer kontinuerligt nya tekniker som öppnar upp sätt att kommunicera, två av dessa är smarta mobiltelefoner och smarta klockor. Google har nu presenterat en bärbar produkt, *Google Glass*, glasögon med en skärm monterad nära ögat för att ge användaren en illusion av att det flyter en skärm framför dem. Eftersom tekniken fortfarande är i ett tidigt skede är inte användningsområdena fastställda och många frågar sig nu vad tekniken kan användas till.

Att smarta glasögon är någonting nytt på marknaden gör att dess funktionalitet jämförs med smarta telefoner och laptops, när det egentligen är en helt ny produktkategori. Det leder också till att användare som är vana vid telefoner och datorer kommer att behöva ändra hur de interagerar med mjuk- och hårdvara i de smarta glasögonen, för att anpassa sig till den nya tekniken.

1.1 Motivering till projekt

Antalet medarbetare på konsultföretag växer och därför behövs kontinuerligt fler kontorsplatser för att alla medarbetare ska få plats, vilket i sin tur kräver både mer administration och större kontor. Sigma IT/Management i Göteborg har valt att använda sig av ett kontorskoncept kallat *open workspace* med *clean desk policy,* som gör det möjligt att ha fler medarbetare i ett mindre kontor. Open workspace med clean desk policy betyder att det finns få fasta placeringar, *”En konsult ska inte vara på kontoret, utan hos ett företag”* [1]. Det gör att Sigma har möjlighet att spara yta genom att ha färre kontorsplatser än medarbetade och de låter därför sina medarbetare välja en ny arbetsplats varje dag. Det har också öppnat för att ha fler mötesrum eftersom mindre yta används till arbetsplatser.

Att investera i nya kontorslandskap är en stor utgift för många företag, därför vill företagen få en visning av Sigmas kontor i Göteborg för att skapa en egen uppfattning av konceptet. Sigma är dessutom ett företag som ligger i framkant inom teknisk konceptutveckling och applicering och vill därför gärna visa upp sina kontor och framförallt de smarta glasögonen för samarbetspartners och potentiella kunder.

Mycket tid har visat sig behöva avsättas av anställda på Sigma för att hålla rundturer inne på kontoret. Det är svårt att hålla informationen konsekvent eftersom det ofta är olika anställda som håller i rundturerna. Sigma vill därför utforska möjligheten att ge besökare en rundvandring av deras kontor i Göteborg genom att använda Google Glass, vilket kan spara både tid och pengar för företaget. Informationen besökare får kommer, genom den nya tekniken, att vara konsekvent under rundvandringarna och kan uppdateras vid behov.

1.2 Syfte

Syftet med projektet är att skapa en applikation för Google Glass som ger besökare en rundtur på Sigmas kontor i Göteborg utan att personal aktivt behöver avsättas för en sådan. Applikationen ska samtidigt visa vilka möjligheter Google Glass har och hur företag kan använda sig av dessa för att effektivisera sin verksamhet. Applikationen behöver en tillhörande webbplats och databas för att skapa och spara informationen som glasögonen ska visa upp.

Google Glass passar bra för uppgiften eftersom informationen kan presenteras likt ett bildspel med text framför användaren. Glasögonen används på huvudet och ser det användaren ser, därför kommer det att kännas naturligt för användaren att titta på ett rum samtidigt som det beskrivs, medan tillhörande bilder om lokalytan presenteras.

Att smarta glasögon är en ny produktkategori kan vara negativt. Nya användare kommer inte att vara bekväma med att använda det nya gränssnittet och kan därför behöva hjälp för att förstå hur applikationen fungerar. Därför behövs en applikation som är lättförstådd och självförklarande.

1.3 Akademisk frågeställning

För att kunna pröva rapportens akademiska frågeställning måste systemet innehålla vissa komponenter. Det krävs ett sätt att lagra och underhålla information samt en applikation till Google Glass. Genom dessa krav kan systemet svara på följande frågeställning:

- Hur kan ett system med smarta glasögon utformas för guidning i kontorslandskap?
- Vad krävs av applikationen för att en förstagångsanvändare ska kunna förstå hur hård- och mjukvaran används?
- Hur jämför sig en rundvandring där guidning sker med smarta glasögon med en rundvandring med mänsklig guide?

1.4 Avgränsningar

Applikationen kommer att vara på engelska. Det är fullt möjligt att översätta information till olika språk genom Google Translate API [2] i glasögonen. Att formatera texter så att de blir korrekt översatta, samt implementera funktionen, ligger dock utanför projektets tidsramar.

Informationen som glasögonen visar kommer att begränsas till 330 tecken för texter. Det går att skapa lösningar som möjliggör att visa mer text på en vy i glasögonen, men även det ligger utanför projektets tidsramar.

Ett begränsat antal personer som använt Google Glass för en rundtur på Sigma svarar på ett frågeformulär kring användandet av glasögonen, som kommer att ligga till grund för projektets slutsatser.

Jämförelsen mellan glasögonen och den mänskliga guiden kommer att avgränsas till 17 personer. Åtta personer kommer att få en vanlig rundvandring på kontoret med en mänsklig guide, de resterande kommer, precis som nämnts, få en rundvandring med hjälp av de smarta glasögonen. Båda parterna kommer därefter att svara på ett formulär där information om olika ytor efterfrågas. De som använt glasögonen kommer därför att fylla i två formulär, medan övriga endast fyller i ett.

1.5 Disposition

Rapporten inleds med att beskriva den metod som applicerades under projektet, därefter kommer de tekniska bakgrundskunskaper som krävs för att förstå innehållet samt motiveringen till de tagna besluten att presenteras. Efter det presenteras tillvägagångssättet som användes under projektet. Genomförandet och resultaten presenteras därefter. Slutligen kommer en diskussion och slutsats kring projektets resultat att redovisas.

2 Metod

Arbetet delas upp i tre olika faser, utveckling av *databas*, *webbplats* och *applikation*. Projektet inleds med utveckling av databasen eftersom dess funktionalitet är kritisk för de resterande momenten. Databasen behöver lagra information om bilder och texter för att senare kunna visas i glasögonen. Utformningen av databasen måste vara väl genomtänkt så att den inte behövs byggas om vid ett senare skede under projektet och den måste vara lätt att bygga ut om extra funktioner krävs.

I nästa steg utformas hemsidan, fokus ligger på att skapa en funktionell produkt istället för en estetisk design. Webbplatsen skapas för att berika databasen med information och för att en administratör ska kunna skapa en rundvandring med hjälp av ett lättanvänt grafiskt gränssnitt. Webbplatsen ska klara av att svara på *HTTP-Get* förfrågningar, vilket är till stor nytta i utvecklingen av applikationen, eftersom information från databasen behövs.

Sista steget i projektet är att utveckla applikationen till Google Glass, eftersom informationen som ska visas är direkt beroende av de två ovan nämnda momenten. Glasögonen ska hämta information från databasen genom att veta var besökarna befinner sig och därefter skicka förfrågningar om information kring lokalytan.

Under projektets slutskede besöker två olika grupper Sigma, där den första gruppen får en guidad rundtur, för att efteråt svara på ett formulär. Formuläret innehåller informativa frågor om olika ytor i kontoret. Den andra gruppen svarar på samma frågor men får rundvandringen med Google Glass som guide och de svarar på ett ytterligare formulär om hur de upplevde rundvandringen med glasögonen.

Under projektet används arbetssättet *Scrum* [3] som använder sig av s*prints*, en *product owner*, *Scrum master* och ett *development team*. Under projektet agerar representanter från Sigma Scrum master och product owner, även om författarna i stor utsträckning har möjlighet att påverka vad definitionen av *done* och *done&done* innebär. Varje sprint fortlöper under två veckor. En specifik sprint *backlog* definierar vilka uppgifter som ska utföras under den givna sprinten.

3 Teknisk Bakgrund

Följande kapitel behandlar tekniska bakgrundskunskaper som krävs för att läsaren ska förstå varför de olika tekniska lösningarna valdes, samt förklara vad dessa innebär.

3.1 ASP.NET

ASP.NET är en kombination av programmeringsspråket ASP, Active Server Pages, och biblioteket .NET Frameworks. ASP är ett språk som körs på webbservern och sammanfogar HTML, script och vanligtvis C# eller Visual Basic kod, för att leverera interaktiva hemsidor till användaren [4].

3.2 Razor

Razor är ett tagg-språk som fungerar liknande HTML, men som också möjliggör att bädda in kod som C# och Visual Basic direkt i HTML-kod. Razor-kod exekveras på serversidan för att sedan leverera HTML-kod till klientens webbläsare [5][6].

3.3 SQL och MSSQL

SQL står för Structured Query Language och är primärspråket för att kommunicera med databaser, samt att ge den struktur som efterfrågas. MSSQL är Microsofts version av en SQL-server som är utvecklad för att fungera bra och smidigt tillsammans med ASP.NET-språket och dess plattform [7].

3.4 Model-View-Controller

Model-View-Controller, MVC, är från grunden ett designmönster som används för att separera olika funktioner i ett system, samt att ge fler lager som bidrar till att designen blir lättförstådd [8]. MVC har även använts för att skapa ett ramverk i Visual Studio för att med ASP.NET kunna skapa applikationer med en tydlig och enhetlig struktur [9].

3.5 Google Glass Development Kit

Glass Development Kit, GDK, är det som används för att utveckla applikationer till Google Glass. GDK är ett tillägg till Android [10] och innehåller extra funktionalitet, till exempel röststyrning, gest-styrning och live-cards [11]. Google GDK är designat för att ge utvecklare en direkt tillgång till hårdvaran som finns tillgänglig.

3.6 QR-kod

QR står för Quick Response och fungerar som en tvådimensionell streckkod där information kan avläsas oavsett orientering. QR-koden utvecklades av Denso Wave år 1994 när vanliga EAN-koder blev för klumpiga och krävde för mycket

plats. Eftersom EAN-koderna är större än QR-koder krävs det mer bläck för att tryckas och därför blir de dyrare [12].

En QR-kod kan läsas både horisontellt och vertikalt, detta med hjälp av tre statiska fyrkanter, en i både det översta vänstra och högra hörnen, samt en i det nedre vänstra hörnet. Dessa fyrkanter hjälper scannern att veta orienteringen av koden när den läses av [12].

En QR-kod kan lagra ca 4000 bokstäver och klarar av att visa all data även om upp till 30% av koden är förstörd, vilket är betydligt mer än EAN-koder klarar av [12].

3.7 GPS -lokalisering

GPS står för Global Positioning System [13] och utvecklades av Amerikanska försvaret för positionering av militära fordon och navigering för robotar [14]. GPS använder sig av satelliter för att kunna bestämma positioneringen, satelliterna skickar signaler med aktuell tid och plats i omloppsbanan som GPS mottagare tar emot. Mottagaren beräknar sedan aktuell position genom att jämföra tiden signalerna sändes. Precisionen som kan utvinnas av en GPS kan bero på flera olika faktorer, om mottagaren är inomhus eller utomhus, atmosfären, mottagarens kvalitet med mera [15].

4 Tillvägagångssätt

Detta kapitel inleds med en presentation av den arbetsmetodik och de verktyg som användes i projektet för att arbetet skulle fortlöpa smidigt och enligt de satta tidsramarna. Därefter kommer designvalen för databasen att presenteras för att övergå till de olika tekniska lösningar projektet applicerar på webbplatsen. Slutligen kommer de olika tekniska lösningarna för Google Glass applikationen att presenteras.

4.1 Projekt

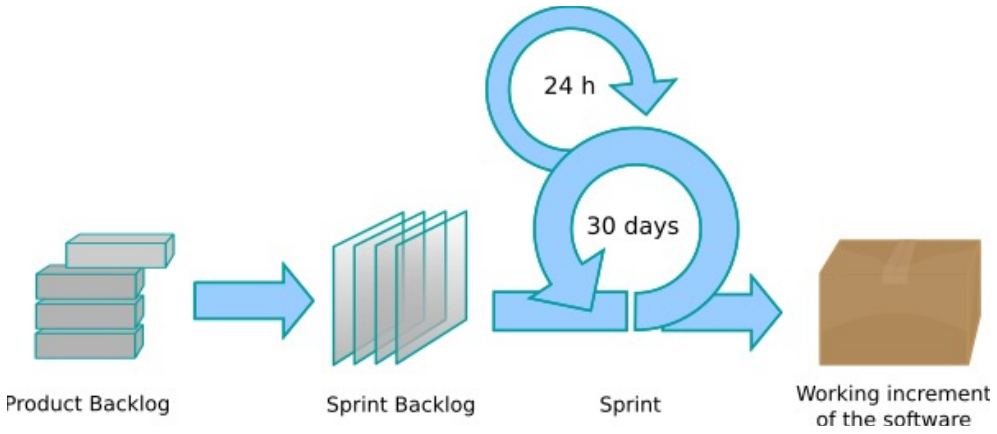
Projektet har använt sig av Scrum, vilket är ett flexibelt arbetssätt för mjukvaruutveckling som många stora företag använder sig av [16]. Projektet implementerar Trello [17], ett webbaserat verktyg som underlättar det administrativa med Scrum, på förslag av Sigma.

4.1.1 Scrum

Scrum är ett *agilt* arbetssätt där arbetet delas upp till korta intervaller, två månaders utveckling kan delas upp till sex olika sprintar. Varje sprint har en backlog där alla de uppgifter som ska utföras innan sprintens slut är dokumenterade, en *implementing* kategori där utvecklare producerar kod och till sist finns kategorin done. Done, enligt Scrum, är när product owner eller Scrum master är nöjd med slutprodukten och den uppfyller alla de ursprungliga kraven. Många Scrum-projekt brukar även implementera done&done, vilket är när product owner och Scrum master båda är nöjda, det existerar inte några buggar och mjukvaran är integrerad i slutprodukten.

Under projektets gång har backlog, *bugs*, *development*, *testing*, *checked in*, *code review* och done använts, samt *sprint review* och *sprint planning*. Dessa arbetsmetoder hjälpte projektet att bli klart inom de satta tidsramarna och gav en fungerande modul i slutet av sprintarna, vilket i sin tur hjälpte integreringen av funktionen i nästkommande sprint.

Fördelen att använda Scrum jämfört med andra metoder är att det efter en sprint finns en fungerande modul för mjukvaran som visas för product ownern, som kom med synpunkter och tips på förbättringar [18][19].



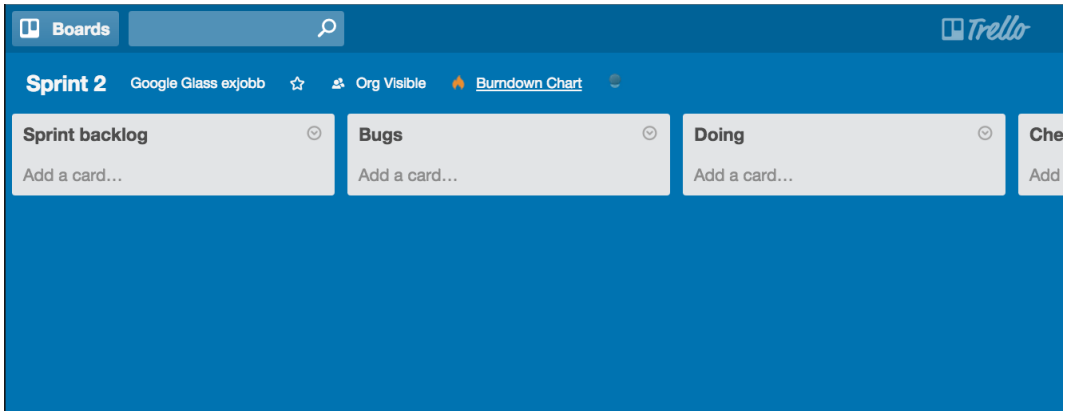
Figur 4.1 Illustration av Scrum-prosessen[20]

Projektets sprint backlog producerades av en product backlog, också kallad *user stories,* se figur 4.1. Eftersom Sigma agerade product owner och Scrum master fanns redan user stories när projektet startade, vilka senare fördes in i projektets backlog.

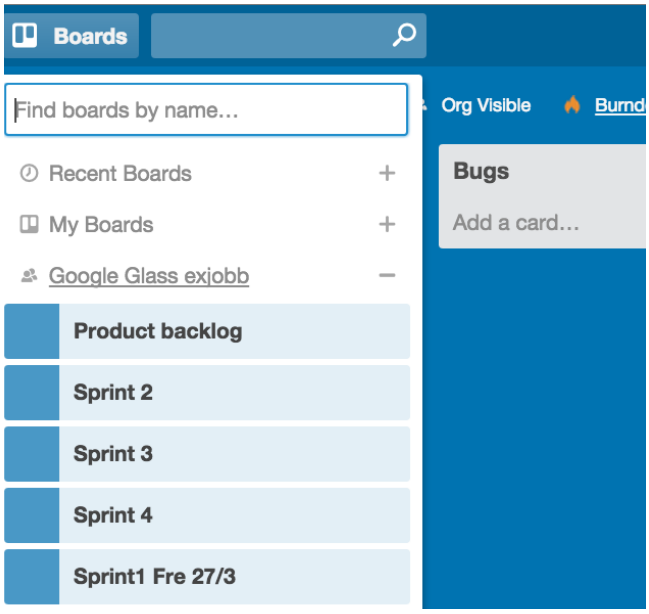
4.1.2 Trello

Trello är ett webbaserat verktyg för att hantera information om backlog och sprintar och underlättar det administrativa arbete som annars krävs för att Scrum- och sprintplaneringen ska fungera smidigt [18].

Varje sprint planerades på Trello med hjälp av ett *Scrumboard,* se figur 4.2 och 4.3, där utvecklarna kan ta på sig uppgifter i former av kort och meddela övriga utvecklare att uppgiften har blivit åtagen. När utvecklaren anser uppgiften slutförd kan kortet flyttas till code review, om koden behöver bli granskad. När ett kort ligger i code review skall funktionaliteten kortet avsåg testas. Om buggar upptäcktes under testingen skall kortet flyttas till kategorin bugs, med en beskrivning av vad buggen var. Om koden är felfri flyttas den till checked in och är då redo att implementeras.



Figur 4.2 Trello's webbgränssnitt för Scrumboard [17]



Figur 4.3 Trello's webbgränssnitt för hantering av Scrumboard [17]

När en funktion blev klar under en sprint lades den under kategorin done, vilket markerade att funktionaliteten kortet avsåg var färdigställt och inte behövde utvecklas ytterligare under nästa sprint.

4.1.3 Projektuppdelning

Utvecklingen delades upp i tre olika faser, som nämnts i kapitel 2, vilket var en stor fördel eftersom det gjorde att författarna endast behövde fokusera på ett moment åt gången. Den första delen som genomfördes var design och utveckling av databasen eftersom dess funktionalitet var kritisk för att de övriga momenten skulle fungera.

Nästa del som utvecklades var en webbplats för att databasen skulle kunna berikas med information en användare angav. Under utvecklingen av webbplatsen kunde även fel i databasstrukturen uppmärksammas och lösas. Webbplatsen skulle även implementera en funktion för att hämta information från databasen genom HTTP-get anrop, eftersom det underlättar kommunikationen mellan glasögonen och databasen.

Det sista steget i projektet var att utveckla applikationen till glasögonen och implementera den med de tidigare faserna. Eftersom den information glasögonen behövde åtkomst till redan existerade i databasen, vilken hemsidan hade åtkomst till, blev implementeringen av funktioner enkel och smidig.

För att utvärdera applikationen krävdes ett frågeformulär som kunde mäta hur mycket information besökaren uppfattat. Formuläret gavs till två grupper som besökte Sigma, den första gruppen fick rundvandringen med en person och och den andra gruppen fick

en rundtur med Google Glass. Som nämnts tidigare fyllde även de personer som använde Google Glass i ett formulär angående användningen av glasögonen.

4.2 Utvecklingsmiljö

Projektet inkluderade en databas, hemsida och applikation vilket ledde till att det krävdes olika utvecklingsmiljöer, nämligen Visual Studio [21] och Andriod Studio [22].

Visual Studio användes vid utveckling av hemsidan och databasen, på förslag av Sigma. Eftersom Visual Studio är den utvecklingsmiljö Sigma använder vid utveckling av mjukvara till kunder. Visual Studio är också ett kraftfullt verktyg för ASP.NET. Sigma kunde genom sin kompetens assistera med support kring utvecklingsmiljön.

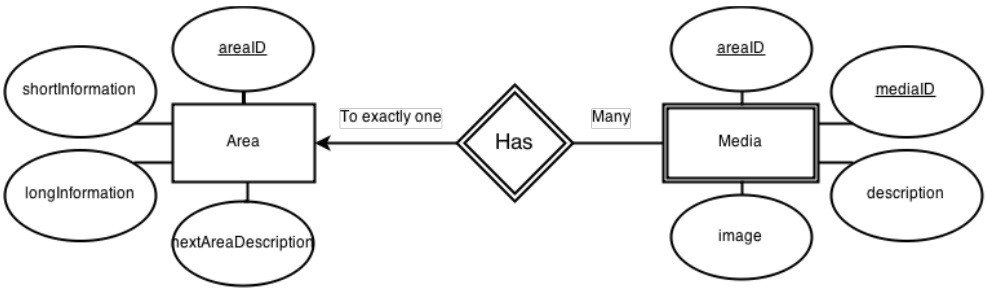
Andriod Studio är Googles egna utvecklingsverktyg och användes för utveckling av mjukvara till Google Glass. Författarna var sedan tidigare bekanta med utvecklingsmiljön.

4.3 Databas

Strukturen på den information som skulle lagras för varje arbetsyta i rundvandringen bestämdes till: titel, kort och lång text, en kort informativ beskrivning till nästa intressanta lokalyta, samt bilder.

Databasen uppdelades till två tabeller eftersom den innehöll lite information. Tabellerna blev Area och Media, där Area innehöll information om lokalytan, medan Media innehöll ett Area-objekt, bild och information om bilden. Area-objekt ska klara av att vara länkad till ett obestämt antal bilder. Media-objekt är en *weak entity* som endast kan existera med hjälp av ett Area-objekt, se figur 4.4.

Att Media är en weak entity av Area kunde nyttjas vid hämtning från databasen, eftersom bilder med samma Area *key* kunde hämtas separat från ursprungshämtningen av Area-objektet. Med denna metod kan alltså en Area länka till flera olika Media-objekt och underlätta hämtningen av bilder till applikationen.



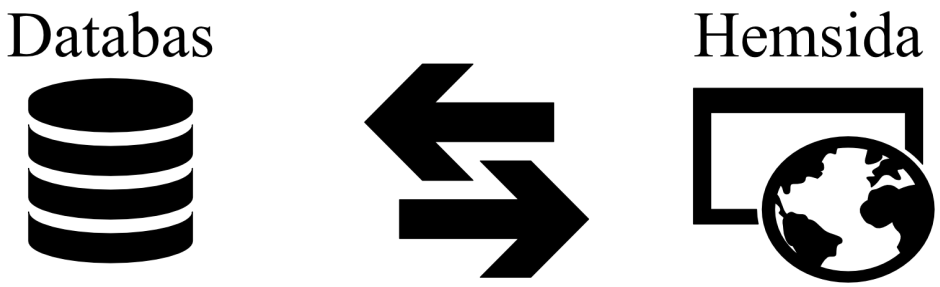
Figur 4.4 Förenklat ER-diagram över databasen

4.4 Hemsida

Hemsidan som skapades för projektet är endast tillgänglig för administratörer, d.v.s. för att komma åt administreringsverktygen krävs inloggning. Hemsidan skapades för att en administratör enkelt ska klara av att berika databasen med relevant information, samt för att kunna redigera redan existerande rundturer.

Samtidigt som en ny beskrivning av en arbetsyta på hemsidan skapas kan användaren välja att ladda upp ett obegränsat antal bilder som automatiskt lagras som Media-objekt, de blir i sin tur kopplade till den specifika Arean i databasen.

Hemsidan klarar också av att svara på HTTP-Get anrop, vilket leder till att specifika Area- och Media-objekt kan returneras från databasen med hjälp av JSON. Att hemsidan klarar av att svara på dessa anrop blev speciellt användbart för applikationen eftersom ingen direkt koppling mellan den och databasen behövdes, hemsidan kan agera och hämta information via anrop, se figur 4.5.



Figur 4.5 Kommunikationen sker endast mellan databasen och hemsidan

4.5 Applikation

Applikationen i Google Glass var tvungen att känna av var någonstans besökspersonerna befinner sig i kontorslandskapet. Detta för att visa upp den relevanta informationen kopplad till den arbetsytan. Eftersom datan som existerar i databasen är direkt kopplad till hemsidan krävs ingen direkt koppling mellan glasögonen och databasen, se figur 4.6. När information om en lokalyta har presenterats skall användaren aktivt välja om mer information behövs genom att ange ett röst- eller touchkommando.

Eftersom smarta glasögon fortfarande är i ett tidigt utvecklingsskede känner inte många till hur navigation i menyer och röstkommandon fungerar. Applikationen hjälper därför användaren utan mänsklig interaktion, detta med hjälp av menyer som är självförklarande och ett enkelt gränssnitt. Gränssnittet följer Google Glass design guidelines [23] för att förenkla för användaren, vilket också var en rekommendation från Sigma.

Figur 4.6 Kommunikationen mellan projektets moduler

5 Genomförande

I detta kapitel beskrivs olika designval och det går även in på hur dessa designval implementerades och hur alla moduler är sammankopplade.

5.1 Designval

Här nedan kommer designvalen för databasen, hemsidan och applikationen att redogöras samt att argumenteras för.

5.1.1 Databas

För informationshantering i Google Glass finns två alternativ, det första är att använda sig av en lokal databas på enheten och det andra är att lägga över ansvaret på en extern server.

Det finns många fördelar med att använda en lokal databas, Google Glass och främst Android har ett brett stöd för att lagra information på enheten [24], och för att den information som sparas är snabbt åtkomlig i en lokal databas. Vid användning av en lokal databas behöver inte enheten tillgång till Internet för anslutning eller hämtning av information, all data finns sparad på enheten.

Nackdelar med att spara information i en lokal databas är att det finns en begränsad mängd lagringsutrymme, 12GB [25]. Eftersom rundturerna som skapas kan variera i storlek blir det svårt att garantera att ledigt utrymme existerar på enheten. En annan nackdel är att det krävs att applikationen uppdateras för att databasen ska uppdateras.

En server ska möjliggöra lagring av informationen på en extern enhet och kommer inte att ta någon plats i minnet på Google Glass, vilket leder till att en större mängd information kan sparas och hanteras utan att lagringsutrymmet på enheten påverkas. Det öppnar också upp för möjligheten att uppdatera informationen från andra enheter, till exempel genom en hemsida.

Att ha informationen på en server har två nackdelar. Den första är att det tar längre tid att utveckla eftersom två separata system måste vägas in. Den andra är att om enheten förlorar uppkopplingen mot serven kommer applikationen att bli oanvändbar.

Under projektet kom en serverlösning att implementeras, anrop skickas till en server där informationen finns lagrad. Valet att använda en server skedde av två huvudanledningar. Den första för att Google Glass har lite lagringsutrymme och det inte är säkert hur mycket information som behöver lagras. Den andra är att det ska vara lätt att vidareutveckla mjukvaran för att anpassas till fler användningsområden.

Under projektet hade också glasögonen tillgång till ett Wi-Fi nätverk eftersom de endast användes på ett kontor. Det säkerställde att glasögonen alltid kunde kommunicera med servern.

5.1.2 Hemsida

Examensarbetet genomfördes hos Sigma som efterfrågade att hemsidan utvecklades i ASP.NET. Företagets kunskaper om programmeringsspråket är stora, det gör det enklare för Sigma att i framtiden vidareutveckla mjukvaran. Sigma kunde därför bistå med hjälp under projektets gång och implementering av databasen till deras servrar.

ASP.NET var inte optimalt för att få den bästa slutprodukten eftersom författarna inte utvecklat i språket tidigare. Eftersom ASP.NET och C# båda är mycket använda språk inom programmering såg dock författarna en möjlighet att lära sig grunderna i de båda språken.

5.1.3 Applikation

När applikationen till Google Glass skulle skapas fanns tre olika alternativ gällande hur glasögonen skulle känna till var de befann sig, samt hur information skulle presenteras.

Det första alternativet var att placera ut QR-koder [26]. Koderna skulle behöva innehålla information så att enheten kunde göra ett anrop till servern för att få ut den relevanta informationen som var kopplad till platsen. Glasögonen letar aktivt efter koder och när en kod upptäcks presenteras informationen om platsen.

Nackdelen med QR-koder är att de fysiskt måste placeras ut på den plats som applikationen ska brukas på. En generator för QR-koder måste också läggas till, eftersom informationen i databasen måste kopplas till koderna.

Det andra alternativet är att enheten aktivt sökte efter vissa mönster eller bilder för att identifiera var någonstans glasögonen befinner sig. Praktiskt skulle det fungera likadant som QR-koder, skillnaden blir vad glasögonen reagerar på. Bildigenkänning skulle varit svårt att realisera eftersom det inte fanns ett lika utbyggt stöd för detta hos Google Glass eller tredjepartsutvecklare.

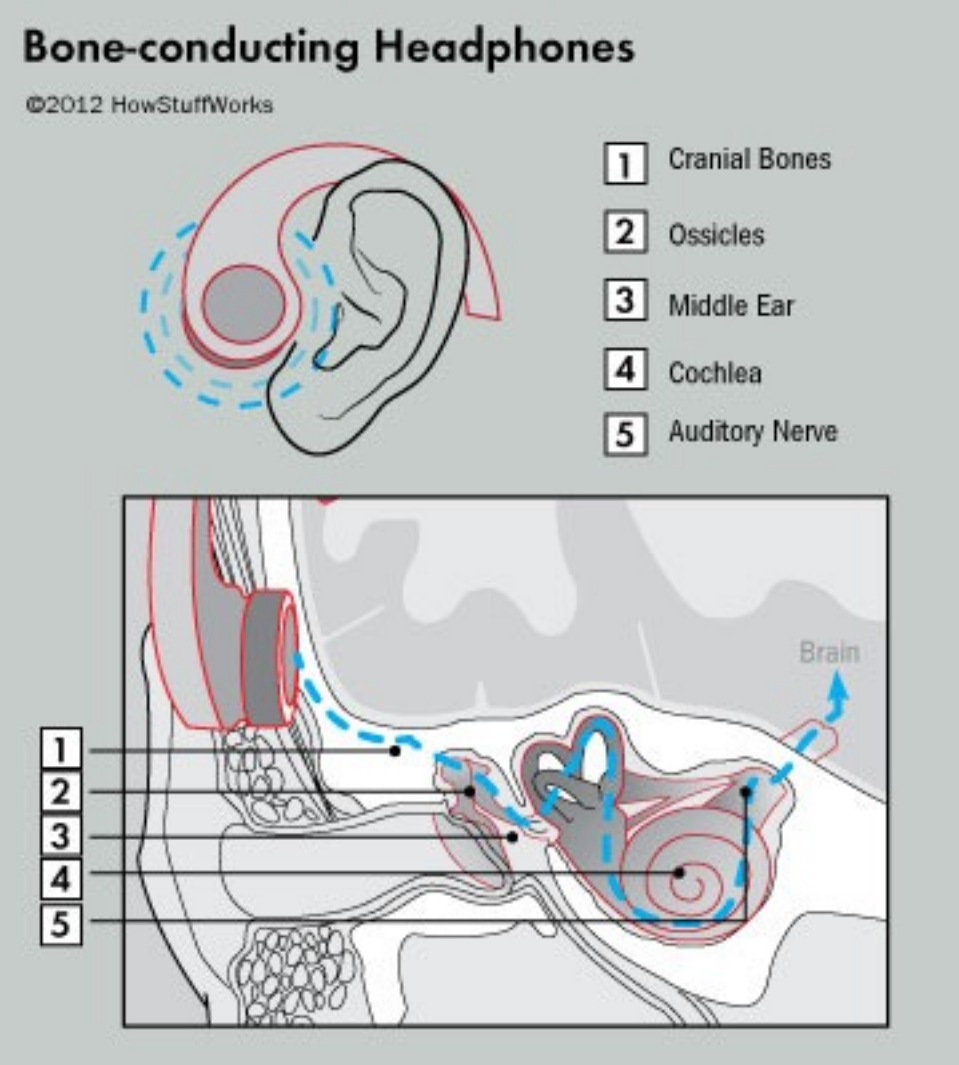
Det sista alternativet var att glasögonen kände till vilken GPS-position de befann sig på och därigenom visade information kopplad till platsen. Google Glass har ingen inbyggd GPS-mottagare vilket ledde till att glasögonen skulle vara tvungna att kompletteras med en extern enhet som har en mottagare, exempelvis en mobiltelefon. Om enheten skulle användas inomhus kunde signalstyrkan vara dålig, vilket kan leda till att enheten inte uppdaterar informationen inom rimlig tid.

Författarna valde att enheten skulle identifiera sin position med hjälp av QR-koder, beslutet togs på grund av att lösningen skulle bli enkel och skalbar. Det finns också ett flertal *open-source* QR-läsare [27] som kan implementeras med Google Glass.

Enheten kan presentera information till användaren på flera olika sätt, där varje sätt har för- och nackdelar. Valet låg mellan två olika alternativ, text eller tal. Att presentera text-information om olika arbetsytor gör att användaren kan välja vilken hastighet rundturen sker på genom att denne läser texten själv, medan uppläsningen rullar på i en bestämd takt.

Skärmen på Google Glass uppfattas som 25” från ca 2,5m avstånd [25]. Det innebär att texten blir svår att läsa om inte den är skriven i ett stort och tydligt typsnitt. Att texten är stor leder i sin tur till att väldigt lite information kan presenteras utan att användaren aktivt behöver rulla texten för att läsa vidare, dessutom avråder “Google Glass design guidelines” från att visa användare långa texter då det kan bli påfrestande för ögonen, samt att den blir väldigt liten [28].

Enheten kan läsa upp texter som endast användaren hör, eftersom Google Glass använder sig av BCT, Bone Conduction Transducer [29]. BCT gör att ingen i kontorslandskapet störs av ljudet och att inget hörlurselement behövs. BCT fungerar genom att skicka vibrationer genom skallbenet till innerörat istället för att leda ljudet via hörselgången, se figur 5.1, det gör att även personer med hörselnedsättning kan höra ljudet.



Figur 5.1 Hur BCT möjliggör att ljud uppfattas via skallbensvibrering [30][31]

Om enheten använder uppläsning kommer det att bli svårt för användaren att kommunicera med personer under rundturen, talet pausas endast när texten är färdigläst.

Projektet valde att implementera en kombination av både text och tal, användaren kunde få en text uppläst, medan denne kunde läsa texten. Användaren kan genom denna metod lägga större fokusering på lokalerna och minde fokus på enheten.

5.1.4 Projektets arbetssätt

Innan projektet startade valdes vilket arbetssätt som skulle användas för projektet, samt hur det skulle utföras. Besluten som togs innan projektet startat hade en markant påverkan på hur uppgifter planerades, fördelades och genomfördes.

Projektet har försökt applicera en avancerad nivå av Scrum med all funktionalitet som det innefattar. Den funktion arbetssättet bidrog med har visat sig vara hjälpsamt under arbetets gång. De två veckor långa sprintarna har möjliggjort att en ny komponent alltid funnits färdig vid sprintens avslut. Detta i sin tur underlättade implementeringen av nya komponenter, samt gav Sigma och författarna möjligheten att utvärdera det som utvecklats.

Projektet skulle vara genomförbart utan Scrum, men vara mycket svårare att färdigställa och skulle det bli svårt att utvärdera och sammankoppla komponenter som utvecklats. Enhetligheten skulle också försämras avsevärt eftersom komponenter skulle utvecklats parallellt istället för sekventiellt. På grund av mycket tillit gavs till arbetssättet kunde författarna fokusera på de uppgifter de åtagit sig, utan att oroas över det som skulle utvecklas och för att produkten i slutändan skulle fungera.

5.1.5 Projektuppdelning

Att projektet blev indelad i tre delar sågs som en självklarhet redan från början eftersom de olika komponenten i projektet var en databas, webbplats och applikation. De tre delarna blev delmål som kunde implementeras separat med nästkommande del i åtanke.

Att databasen utvecklades först gjorde det möjligt att helt fokusera på dess funktionalitet och design. Eftersom författarna endast fokuserade på en del åt gången var kvalitén på koden och dess funktionalitet hög, vilket gjorde utvecklingsprocessen under projektet smidig.

När databasen var färdig startade utvecklingen av webbplatsen vars huvudfunktioner var att berika databasen och svara på anrop från glasögonen. Eftersom webbplatsen hade en direkt koppling till databasen ledde det till att ingen direkt koppling mellan glasögonen och databasen behövdes.

Det sista som utvecklades i projektet var applikationen till Google Glass, men eftersom de två tidigare delarna redan var färdiga kunde koden som producerades hantera verklig data från början.

När utvecklingsarbetet var färdigställt skulle systemets funktionalitet testas genom att besökare fick en rundvandring hos Sigma, med eller utan de smarta glasögonen. Efter rundturen fick de svara på informativa frågor, den grupp som använt Google Glass fick också svara på ett formulär om hur de upplevt applikationen. De informativa frågorna skulle sedan jämföras mellan grupperna för att kunna se eventuella likheter och skillnader i hur mycket information som hade uppfattats. Frågeformuläret för Google Glass utformades för att svara på den akademiska frågeställningen angående användarvänligheten i applikationen.

Att utvecklingen delades upp i tre olika faser visade sig bli bättre än väntat, alla moment fick ett större fokus och blev mer genomtänkta än de annars skulle blivit. Projektet fick också en tydlig struktur.

5.2 Implementering

Innan starten av utvecklingen inhämtades information från Microsoft Virtual Academy [32], för att båda författarna skulle bli bekanta med utvecklingsmiljön som skulle användas. Efter inlärningsfasen stod det klart att databasen kunde skapas från C#-kod, vilket både skulle göra implementeringen av databasen lättare samtidigt som sammankopplingen av databasen och hemsidan blev smidig.

5.2.1 Databas

Utvecklingen av databasen började med att den information som behövde lagras efterforskades för att skapa de nödvändiga tabeller och kolumner som skulle vägas in i designen. När en lämplig design (se avsnitt 4.3) var färdig skapades databasen genom C#-kod och med hjälp av Code First Entity Framework [33][34]. Koden som användes för att skapa ett Media-objekt ses i figur 5.2.

```
namespace GlassServer.Models {
    public class Media {
        [Key]
        public int MediaID { get; set; }
        public int AreaID { get; set; }

        public byte[] Content { get; set; }
        public string ContentType { get; set; }
        public string Description { get; set; }
```

Figur 5.2 Koden som användes för att generera databastabellen Media

Media är en weak-entity av ett Area-objekt, därför är både MediaID och AreaID nycklar till objektet, se avsnitt 4.3. MediaID är en primärnyckel och AreaID en sekundärnyckel i ett Media-objekt. Bilden som lagras konverteras från en HttpPostedFileBase till en byte-array och en sträng C*ontentType*, eftersom bilden måste sparas i bytes. Anledningen till att ContentType behövs är för att avgöra filformatet som bilden använder. Det gör att *encodingen* som används när bilden ritas upp ändras beroende på vilket ContentType som har lagrats för bilden. Att databasen sparar filtypen och känner till hur den är kodad leder till att flera bildformat stöds.

Area-objektet har skapats på samma sätt som Media-objektet. Det som skiljer dem åt är informationen som de lagrar. Värt att notera är att Area-objektet innehåller en lista med MediaID, vilket gör att Area-objektet kan innehålla flera bilder utan att själv behöva hålla informationen om bilderna.

```
namespace GlassServer.Models {
    public class Area {
        [Key]
        public int AreaID { get; set; }

        public string Title { get; set; }
        public string LongText { get; set; }
        public string ShortText { get; set; }
        public int NextAreaID { get; set; }
        public Area NextArea { get; set; }
        public String NextAreaDirection { get; set; }
        public virtual List<Media> medias { get; set; }
```

Figur 5.3 Koden som användes för att generera databastabellen Area

Sigma tillhandahöll en egen databasserver som databasen implementeras på. Databasen skrevs lokalt i Visual Studio och ASP.NET, sedan hjälpte en av företagets utvecklare till med uppladdningen till servern.

5.2.2 Hemsida

Hemsidan har (se avsnitt 4.2) utvecklats i Visual Studio och ASP.NET eftersom Sigma specifikt efterfrågade det och för att projektet lättare skulle kunna implementeras med deras redan existerande server.

Arbetet startades med att generera ett testprojekt med hjälp av Visual Studio, alla komponenter såsom Bootstrap [35], vyer, kontrollers och modeller genererades automatiskt för att projektet skulle ha en stabil grund. När komponenterna hade blivit genererade började vyerna att modifieras för att passa de krav projektet hade.

Den första vyn som skapades var den som ansvarade för att berika databasen med information om Area, de olika ytorna i kontoret. Eftersom datan som skulle lagras i databasen redan var bestämd skapades fält där användaren kunde fylla i den relevanta informationen om det specifika objektet. Vyn returnerade sedan informationen till kontrollen som formaterade informationen och lagrade den som ett nytt objekt i databasen. Area-objekt innehåller en lista över olika MediaID, som den senare ska använda sig av. Denna lista lämnades tom eftersom ingen Media-vy ännu hade skapats.

När den första vyn var modifierad för att berika databasen med relevant information, påbörjades arbetet med att implementera en vy som klarade av att berika databasen med Media-objekt.

När en fil ska laddas upp i ASP.NET krävs att klassen HttpPostedFileBase används, eftersom den innehåller all relevant information om filen. Detta ledde direkt till problem. Eftersom HttpPostedFileBase är ett avancerat objekt som kräver många och stora fält blir det svårt att lagra denna typ av objekt i en databas då den innehåller

mycket onödig information för det specifika syftet. Istället måste den relevanta datan först utvinnas från klassen. På grund av problemet med HttpPostedFileBase krävdes att projektet implementerade en hjälpvy som höll i objektet för att sedan konvertera till något som kunde lagras i databasen. Denna hjälpklass kallas för MediaViewModel (se figur 5.4).

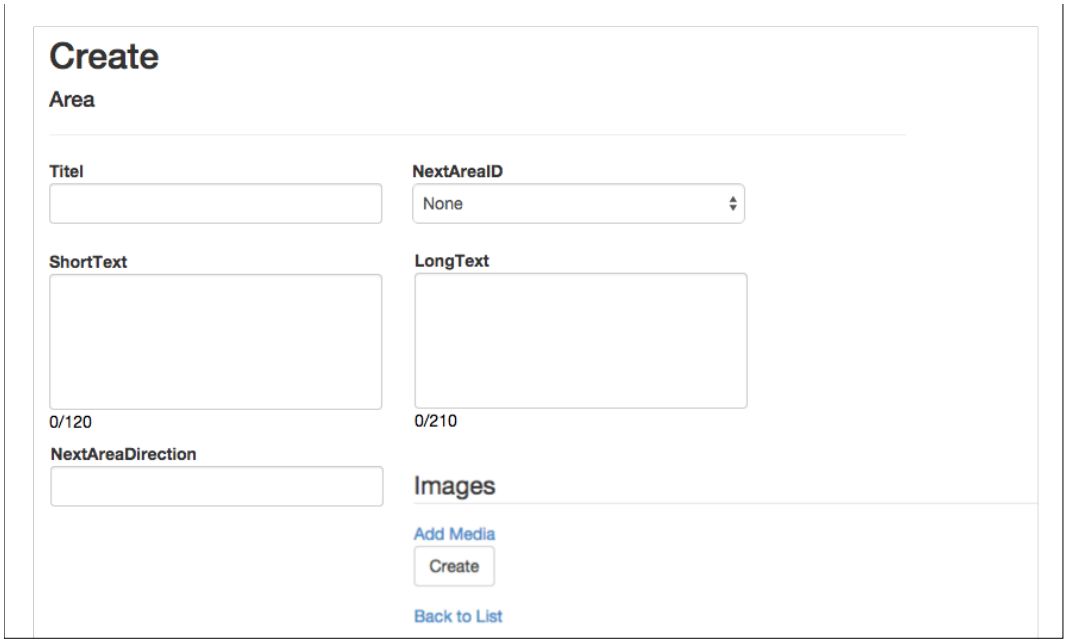
```
namespace GlassServer.Models {
      public class MediaViewModel {
            [Key]
            public int MediaID {get; set;}

            public string Description {get; set;}
            [DataType(DataType.Upload)]
            public HttpPostedFileBase Content {get; set;}
            public Area Area { get; set; }
            public int AreaID { get; set; }
```

Figur 5.4 Koden som som utgjorde hjälpklassen MediaViewModel

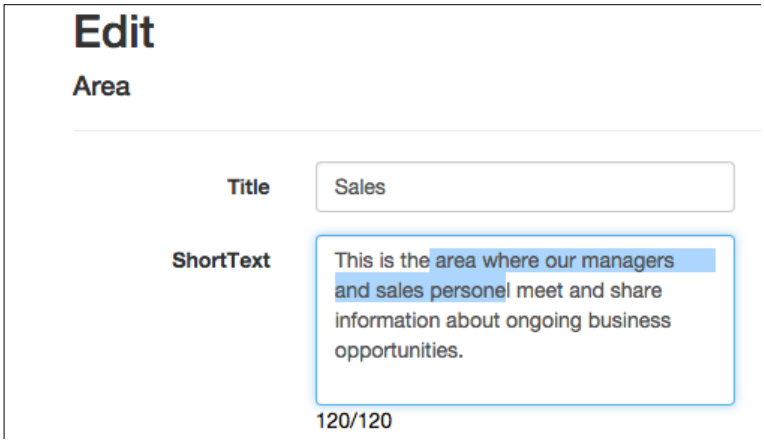
Efter att ett MediaViewModel-objekt skapats, extraheras den information som är relevant för lagring i databasen. Den relevanta information som lagrades var bilden och dess format. Genom att använda hjälpklassen lagras bilder i databasen som en byte-array med ett tillhörande bildformat kallat ContentType (se figur 5.2).

När båda vyerna var klara stod det klart att det var väldigt krångligt att först behöva ett Area-objekt, sedan skapa ett Media-objekt och sist länka Media-objektet till Area-objektet, detta endast för att lägga till en bild för ett Area-objekt. Därför slogs vyerna samman till en kombinerad vy där all information om lokalytan kunde skrivas in på en gång och bilder kunde laddas upp. Bilderna som lades till i den kombinerade vyn blev fortfarande Media-objekt, men blev automatiskt sammankopplade med Area-objektet (se figur 5.5).



Figur 5.5 Sammankopplade vyn för Create Area och Media på hemsidan

När alla vyer för att lägga till ny information var färdiga och fungerade, började arbetet med Edit-vyn som används för att redigera information som redan existerar för en Area. Edit-vyn är nästintill identisk med Create-vyn. Den enda noterbara skillnaden är att fälten som visas skall vara berikade med information från databasen. (se figur 5.6).



Figur 5.6 Redigerings vy, användaren ändrar information i databasen

Eftersom den information som går att ändra redan är ifylld, kan användaren enkelt redigera det som existerar i databasen. Bilderna för en sparad arbetsyta kan också ändras genom att gamla Media-objekt som är kopplade till Area-objektet tas bort och de nya Media-objekten, som användaren valde att ladda upp, kopplas samman med Area-objektet.

När både Edit- och Create-vyerna var färdigställda startade utvecklingen av Detail-vyn. Detail-vyn används för att visa den information och bilder som varje Area innehåller, samt QR-koden för objektet. QR-koden genereras av AreaID-nyckeln för arbetsytan genom ett jQuery-bibliotek [36]. Detta leder till att QR-koden inte innehåller någon information om själva arbetsytan utan bara en siffra, därav kan ingen information utvinnas om lokalytan om någon annan enhet än glasögonen med applikationen skannar koden.

För att en rundvandring ska vara enkel att skapa och underhålla skapades en funktion som gör att en användare kan ladda ner ett *.pdf*-dokument innehållande QR-koden och titeln på Area-objektet, processen kan ses i figur 5.7.

Figur 5.7 Användaren laddar ner en .pdf-fil med QR-koden och titeln för Arean

För att QR-koden ska kunna laddas ner krävs hjälp av ett externt bibliotek kallat Rotativa [37]. Biblioteket skapar en .pdf-fil av en HTML-sida. Det gjorde att en HTML-sida, som kunde konverteras till en nedladdningsbar .pdf, var tvungen att skapas. Eftersom det finns många parametrar för bildstorleken och textens storlek, definierades tre olika alternativ för användaren, small, medium och large. Small ger användaren en liten QR-kod medan large fyller ett helt A4 papper. Large formatet kan ses på QR-koden i figur 5.7.

Den sista funktionalitet som skapades för webbplatsen var att den skulle klara av att svara på HTTP-get och returnera en JSON-objekt. HTTP-get sker med hjälp av en url-parameter till det specifika Area-objektet som returnerar objektet formaterat som en JSON-objekt. Innan objektet returneras itereras alla Media-objekt och för varje objekt ändras byte-arrayen till en tom byte-array, för att bildhämtningen sker separat. Det

objekt som returnerades efterfrågades genom .NET metoden för skapandet av JSON-objekt, se figur 5.8.

```
{
   "medias":[],
   "AreaID":10,
   "Title":"Sales",
   "LongText":"Lång information om Sales.",
   "ShortText":"Kort information om Sales.",
   "NextAreaID":11,
}
```

Figur 5.8 Exempel på ett returnerat JSON-objekt från webbtjänsten

5.2.3 Applikation

Utvecklingen av Google Glass applikationen började med att skapa ett testprojekt i Android Studio, där de olika modulerna som senare implementerades i slutversionen testades. Därefter skapades klasser som skulle efterlikna Area och Media i databasen då dessa konstant krävs av applikationen. Då Area- och Media-objekten i applikationen inte innehöll någon information började utvecklingen av klassen för nätverkskommunikation kort därefter så de kunde berikas.

Klassen som ansvarar för datahämtning implementerar A*sync-Task* för att applikationen ska kunna göra nätverkshämtning i en separat tråd. Att separera trådar för nätverkskommunikation från huvudtråden är också ett krav i Android. Eftersom webbplattformen redan hade en webbtjänst som gjorde det möjligt för ett objekt att returneras från databasen givet ett id för objektet, är den inparameter som krävs för klassens konstruktor en id-sträng. Detta simplifierade strukturen i andra klasser och ger koden bättre läsbarhet.

Eftersom utformningen av Area- och Media-objekt är annorlunda, delades nätverkskommunikationen upp i två separata klasser, där den ena klassen ansvarade för hämtning av Area-strängar och den andra bitmap-bilder [38]. Area- och Media-klasserna använder i sin tur dessa klasser för att hämta den data de behöver. När utvecklingen av klasserna för nätverkskommunikation färdigställdes påbörjades implementeringen av dessa i konstruktorn för Media och Area.

Konstruktorn för Area-objekt i applikationen kräver ett id som inparameter eftersom det krävs av webbplatsen för att ett Area-objekt ska returneras. Väl inne i konstruktorn skickas en HTTP-get förfrågning till webbtjänsten, resultatet sparas som ett JSON-objekt, objektet i sin tur utvinns på den information det innehåller. För att berika Area-objektet med Media-objekt eller bilder den är kopplad med blir listan från JSON-objektet (se figur 5.8) med mediaID genomgången och skapad, se figur 5.9.

```
for(int i = 0; i<jsonArray.length(); i++){
      JSONObject object = jsonArray.getJSONObject(i);
      Media media = new Media(object);
      medias.add(media);
}
```

Figur 5.9 For-loop som berikar arean eller med flera Media-objekt

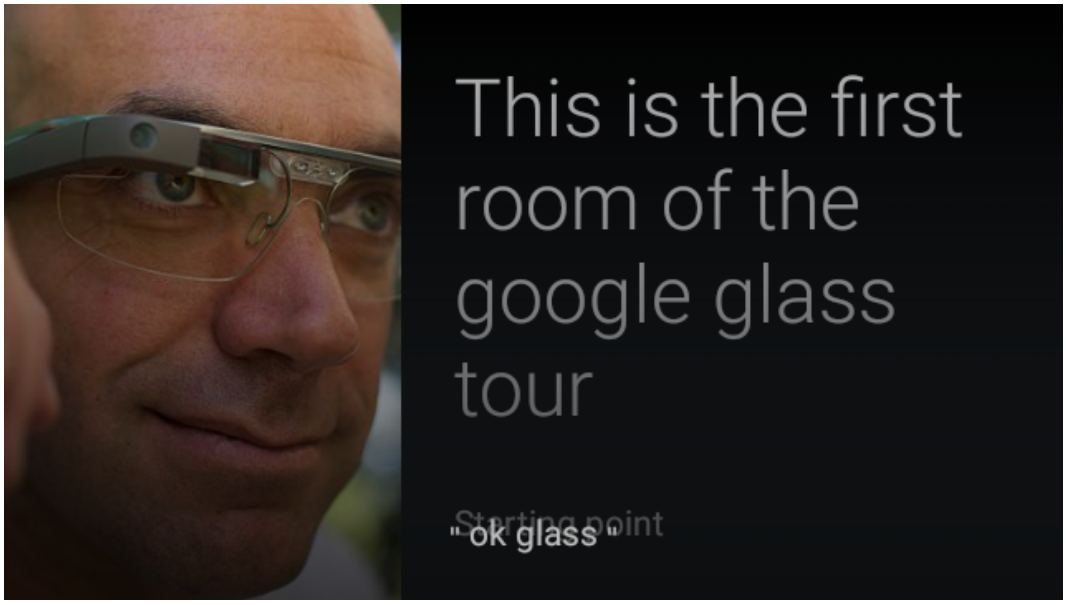
När utvecklingen av de båda objekten var färdigställd påbörjades utvecklingen av de vyer som applikationen innehåller. Det skapades fem vyer för att hantera alla de olika funktioner som applikationen använder: scanner-, shortInformation-, longInformation-, images- och directions-view.

Figur 5.10 scanner-view, vyn visar användaren vad glasögonens kamera ser.

scanner-view möjliggör för applikationen att visa vad de ser i skärmen på glasögonen, se figur 5.10, på det sättet blev det lättare för en användare att skanna QR-koder. När användaren väl har skannat en QR-kod som fanns med i rundvandringen ska applikationen automatiskt hämta informationen som är bunden till det id från databasen. När all information är hämtad och redo att visas presenteras nästa vy för användaren automatiskt, shortInformation-view.

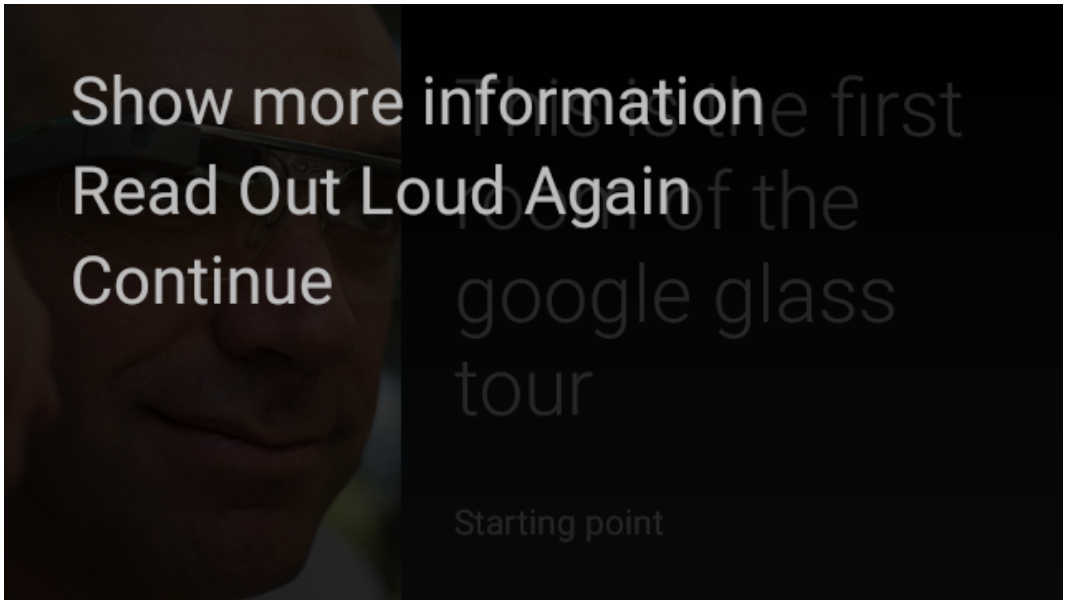
shortInformation-view har som uppdrag i applikationen att visa en kort text i den högra kolumnen och de bilder som är bundna till arbetsytan i den vänstra kolumnen, se figur 5.11. Texten som är bunden till lokalytan ligger lagrad i ett Area-objekt, detta gör att informationen blir lättillgänglig. Den funktion som först började implementeras i denna vy var TextToSpeech som möjliggör att applikationen läser upp textsträngar. Texten

som från början bara visades i den högra kolumnen kunde därmed både läsas av användaren och bli uppläst. När applikationen väl har läst upp texten instruerar den även hur användaren ska agera för att fortsätta rundvandringen genom att berätta de olika alternativ som finns tillgängliga. Nästa funktion som utvecklades var de menyer som användes för att navigera i applikationen, samt för att välja vad som skall presenteras därnäst.



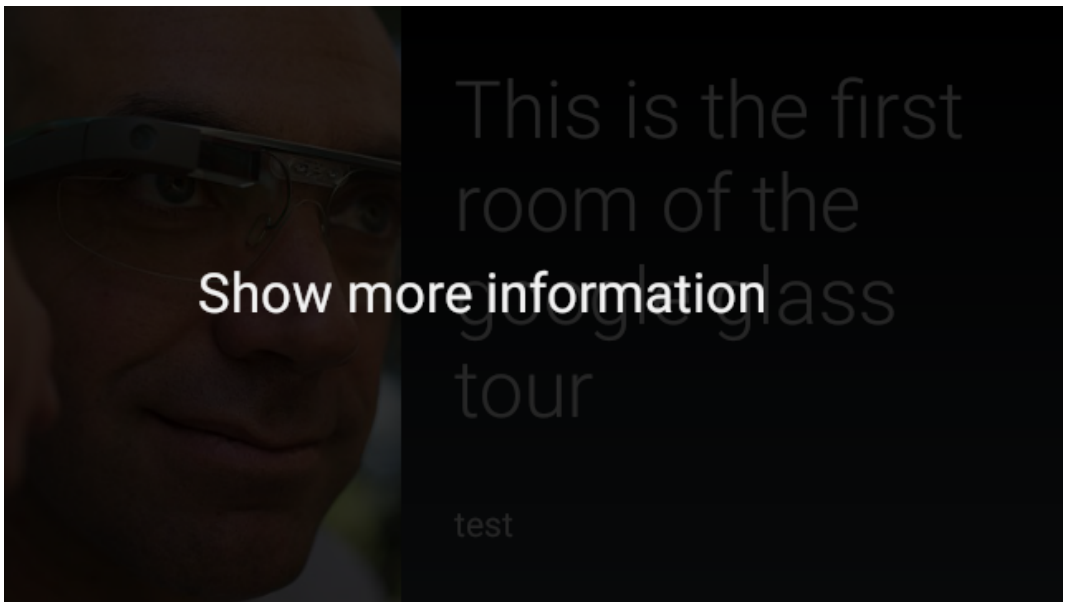
Figur 5.11 shortInformation-view, vyn som användaren ser efter en skannad QR-kod

Författarna ville att två alternativ skulle ges till användaren för navigering i menyer, nämligen röst- och touch-navigering. Röstnavigeringen aktiveras genom att användaren säger: ”ok glass”, för att få en lista presenterad, se figur 5.12, användaren behöver bara säga vilket menyalternativ som skall väljas.



Figur 5.12 Menyns som visa genom röstkommandot ”ok glass”

Det andra alternativet var att användaren skall kunna använda den touch-funktionalitet som finns inbyggd i glasögonen. Genom att trycka på höger sida av glasögonen aktiveras en meny som går att skrolla i, se figur 5.13. För att välja ett alternativ trycker användaren på högersidan. Genom att ge användaren två olika sätt att navigera menyerna hoppades författarna att applikationen ska bli mer lättanvänd. Eftersom ett Area-objekt innehåller både kort och lång information blev nästa steg att designa longInformation-view, en vy som presenterade den långa mer detaljerade informationen som finns lagrad för ett Area-objekt, se figur 5.14. Navigeringen genom menyerna sammankopplas också med den nya vyn i glasögonen.



Figur 5.13 Menyn som visas genom touch-kommando

I longInformation-view blir användaren presenterad den långa mer detaljerade texten om en lokalyta, samt får den uppläst. TextToSpeech implementerades likadant som i shortInformation-view. Att bara texten visas i textvyn för glasögonen, se figur 5.14, är för att mer text än tidigare ska få plats att presenteras utan att textstorleken blir för liten. När texten är uppläst kommer applikationen att automatiskt tala om för användaren hur den ska agera för att fortsätta rundvandringen.

Figur 5.14 longInformationView

Användare som gör det aktiva valet att få mer information om en lokalyta kommer också att få ett nytt menyalternativ, ”show images” ifall bilder finns tillgängliga för lokalytan. Images-vyn kommer då att presentera de bilder som är kopplade till lokalytan (se figur 5.15) genom att låta användaren skrolla igenom den med hjälp av att föra ett finger framåt eller bakåt på glasögonens högersida. När användaren nu väljer att navigera vidare finns bara ett alternativ, att fortsätta rundvandringen.

Figur 5.15 images-view, vyn som används för att presentera bilder för en lokalyta

Den sista vyn som presenteras för användaren är directions-view, se figur 5.16. Denna vy ser precis likadan ut som longInformation-view. directions-view visar endast en text som förklarar hur användaren ska navigera till den nästa intressanta arbetsyta som har en ny QR-kod som kan skannas.

Figur 5.16 directionsView, vyn som beskriver hur nästa QR-kod återfinns

När användaren väljer att scanna en ny kod kommer applikationen att upprepas, likt en sluten cirkel. Författarna har valt att ge successivt färre meny-alternativ då användaren kommer djupare ned i applikationen och till sist presentera det sista alternativet som tar användaren till toppen av applikationen. Detta eftersom användaren inte skall fastna i menyer samt att användaren alltid skall guidas till nästa QR-kod (se appendix A).

5.3 Formulär för undersökning

Den metod som valdes för att svara på frågeställningen om hur glasögonen jämförde sig mot en mänsklig guide, var att genomföra två separata rundvandringar på kontoret. Den första med Google Glass som guide och den andra med en mänsklig guide. Efter rundvandringen fick de båda grupperna svara på ett identiska formulär med frågor på den information de hade fått presenterad under rundvandringen. Gruppen som hade Google Glass som guide fick också svara på ett formulär om hur de upplevde applikationen till glasögonen.

5.3.1 Formulär om rundvandring

Innan rundvandringarna utfördes kontaktades ett flertal personer för att få en uppfattning om hur många personer som var intresserade av att medverka. 17 personer anmälde sig. Åtta personer gick på rundvandringen med en mänsklig guide och nio personer fick rundvandringen med Google Glass.

Frågorna som de båda grupperna fick svara på utformades så att alla svarsalternativen var möjliga, även om det bara fanns ett rätt alternativ som hade presenterats under rundvandringen. Undersökningen fokuserade inte så mycket på de enskilda frågorna,

utan försökte istället analysera de samlade svaren för de båda grupperna. På det sättet kunde slutsatser om jämförelsen mellan glasögonen och den mänskliga guiden dras.

Eftersom rundvandringen med Google Glass innefattade en person mer kom ett formulär att uteslutas, vilket formulär det blev lämnades åt slumpen, detta eftersom urvalsgrupperna skulle innehålla lika många personer. Jämförelsen mellan rundturerna bestod alltså av åtta personer i vardera grupp. Rundvandringen bestod av åtta ytor i kontoret, som alla hade olika information. Formuläret innehöll totalt fem frågor om dessa ytor.

Medelåldern av deltagarna var i 26 år och könsfördelningen för de två grupperna var: sju män och två kvinnor för glasögonen, samt fyra män och fyra kvinnor för den andra gruppen. Eftersom kön och ålder ej angavs på blanketten kan ingen slutsats dras kring om könet eller åldern var relevant i undersökningen.

De deltagande i undersökningen fick verbalt frågan om de provat Google Glass innan, och alla de efterfrågade svarade att de ej hade provat dessa tidigare.

Eftersom den information som gavs till de båda grupperna var konstant var det alltid samma person som presenterade informationen om glasögonen till gruppen som skulle använda dem. Likaså var det alltid samma person som höll i rundvandringen med en mänsklig guide.

De personer som gick rundvandringen med en mänsklig guide tilläts också ställa frågor under och efter rundturen, eftersom det sågs som något självklart att tillåta.

5.3.2 Formulär om Google Glass

Personerna som gick på rundvandringen med Google Glass fick svara på ytterligare ett formulär, efter de hade svarat på det tidigare formuläret kring rundvandringen. Detta formulär innehöll frågor om hur applikationen och dess samverkan med hårdvaran upplevdes. Alla personer som använde glasögonen under rundturen fick också kort information om hur de kunde användas, det för att efterlikna den information som en besökande hos Sigma skulle få innan de själva skulle gå på rundvandringen. Under rundvandringen följde också en av författarna med, då hårdvaran kunde överhettas och var tvungen att startas om.

Eftersom informationen som glasögonen presenterade var konstant, då den inte uppdaterades mellan rundvandringarna, kan slutsatser om hur mjukvaran interagerade med hårdvaran dras.

6 Resultat

I detta kapitel presenteras de resultat som uppnåddes med applikationen, hemsidan och databasen, det vill säga systemlösningen. Därefter presenteras de svar som mottagits på rundvandringsformuläret, samt svaren som mottogs på undersökningen hur förstagångsanvändarna upplevde att mjukvaran på glasögonen interagerade med hårdvaran. Eftersom undersökningen utfördes på engelska har frågorna här översattas till svenska.

6.1 Systemlösning

Under arbetets gång har en databas, hemsida och applikation till Google Glass utvecklats. Projektets mål var att utveckla en systemlösning med hjälp av en Google Glass som då möjliggjorde för Sigma att frigöra de anställdas tid genom att automatisera de rundturer de var tvungna att hålla på kontoret.

6.1.1 Databas

Databasen som har utvecklats löser problemet med att lagra den information som rundturen ska innehålla. Den information som lagras i databasen är en kort respektive en lång informativ beskrivning, bilder kopplade till en arbetsyta och slutligen en beskrivning hur användaren hittar till nästa lokalyta i rundturen där information kan presenteras. En visualisering av databasen finns i figur 4.4.

6.1.2 Hemsida

Hemsidan för systemet används för att kunna skapa och redigera rundturer som i sin tur lagras i databasen. Hemsidan ansvarar också för kommunikation mellan databasen och applikationen. Användargränssnittet som implementeras gör att en administratör klarar av att använda tjänsten med minimala förkunskaper.

6.1.3 Applikation

Fokus har legat på att göra applikationen lättanvänd för att användare av Google Glass, i denna miljö, som ej har använt något liknande tidigare. Användargränssnitt och menyer har förklarats ingående under användning och de val som finns tillgängliga har alltid presenterats för användaren för att förenkla användningen.

Genom att applikationen måste terminera kommer alltid användare att få guidning till nästkommande lokalyta, se flödesschemat i Appendix A. Det ges också två olika alternativ för användaren att interagera med applikationen, touch- och röstkommandon.

6.2 Formulär för rundvandring

Formuläret som avsåg att besvara hur en guidning med smarta glasögon var jämfört med en rundvandring med en mänsklig guide fick följande svar.

6.2.1 Fråga 1

”Under vilken tid på dagen är det tillåtet att använda spelkonsolen?”

Svarsfördelningen: Google Glass fem rätt, den mänskliga guiden sex rätt.

6.2.2 Fråga 2

”Om du är på avdelningen sälj, vad ska du göra om telefonen ringer?”

Svarsfördelningen: Google Glass sju rätt, den mänskliga guiden sju rätt.

6.2.3 Fråga 3

”Varför innehöll mötesrummet du såg under rundturen inte någon teknisk utrustning?”

Svarsfördelningen: Google Glass åtta rätt, den mänskliga guiden åtta rätt.

6.2.4 Fråga 4

”Telefonbåsen var designad med ___ i åtanke?”

Svarsfördelningen: Google Glass två rätt, den mänskliga guiden fem rätt.

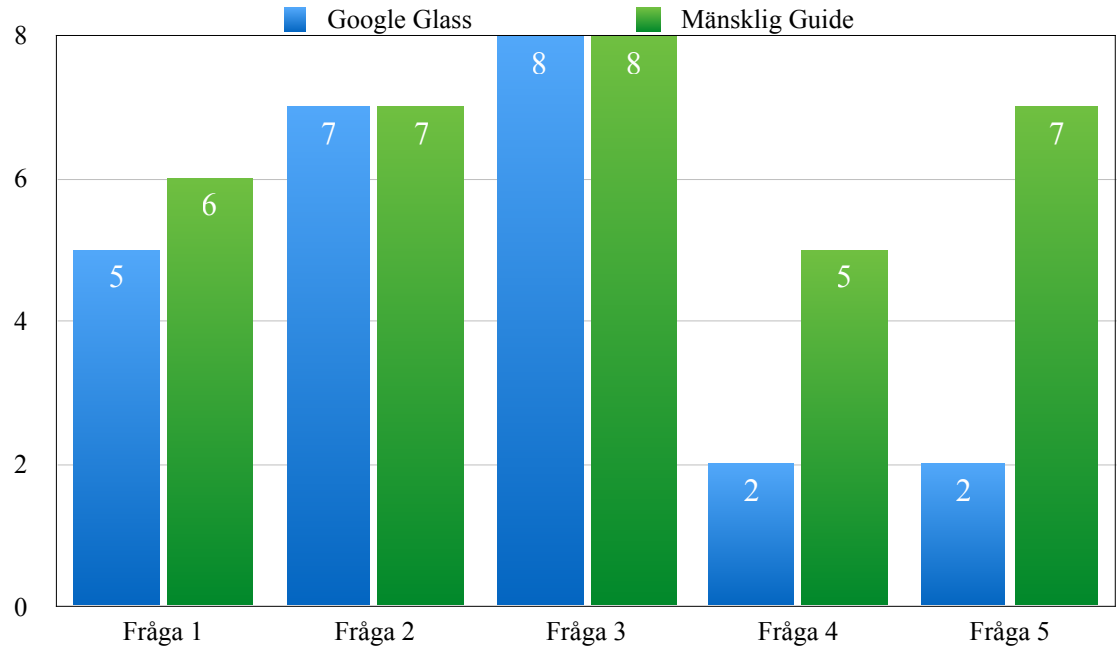
6.2.5 Fråga 5

”Varför finns det ett projektrum på kontoret?”

Svarsfördelningen: Google Glass två rätt, den mänskliga guiden sju rätt.

6.2.6 Sammanställning av rundvandring

Skillnaden mellan antalet rätt valda svarsalternativ för rundvandring med smarta glasögonen och en mänsklig guide återfinns i figur 6.1.



Figur 6.1 Sammanställning av svarsalternativen ur formuläret för Google Glass och den mänskligt guidade rundturen (frågorna är i sekventiell ordning)

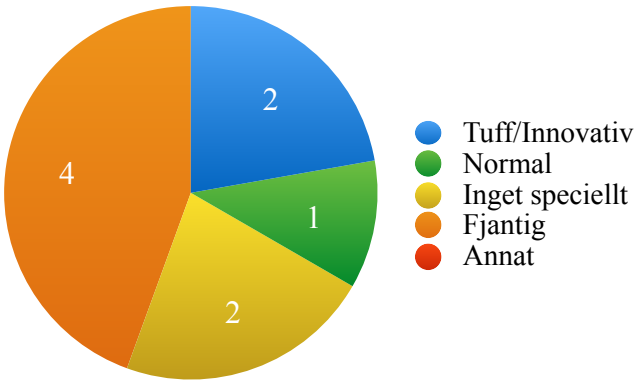
6.3 Formulär för applikation

Eftersom projektets frågeställning var att besvara vad som krävs av applikationen för att en förstagångsanvändare skall förstå hur hård- och mjukvara används, utformades ett formulär som avsåg att besvara frågan. Svaret på formuläret följer nedan.

6.3.1 Fråga 1

”Hur kändes det att ha på sig Google Glass?”

De olika svarsalternativen och fördelningen mellan dessa återfinns i figur 6.2.

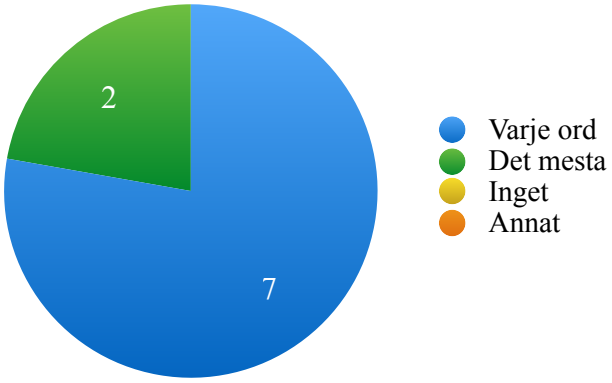


Figur 6.2 Redovisning av svar på fråga 1 ur formulär för applikation

6.3.2 Fråga 2

”Hur mycket av informationen förstod du? (uppskatta)”

De olika svarsalternativen och fördelningen mellan dessa återfinns i figur 6.3.

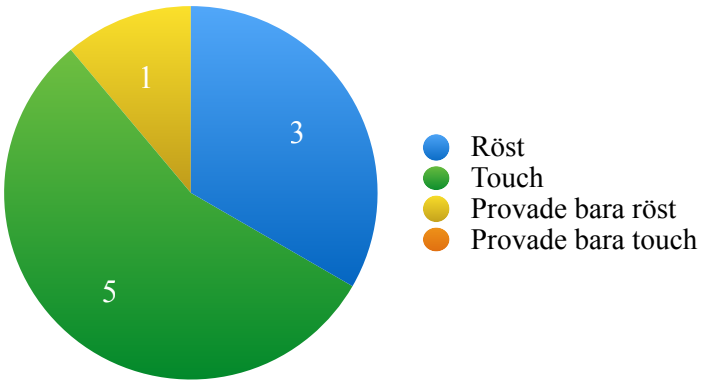


Figur 6.3 Redovisning av svar på fråga 2 ur formulär för applikation

6.3.3 Fråga 3

”Föredrog du att använda röst- eller touch-kommandon?”

De olika svarsalternativen och fördelningen mellan dessa återfinns i figur 6.4.

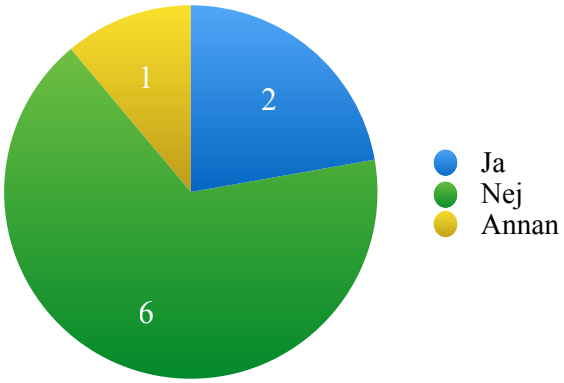


Figur 6.4 Redovisning av svar på fråga 3 ur formulär för applikation

6.3.4 Fråga 4

”Föredrar du guidning med smarta glasögon framför en människa?”

De olika svarsalternativen och fördelningen mellan dessa återfinns i figur 6.5.



Figur 6.5 Redovisning av svar på fråga 4 ur formulär för applikation

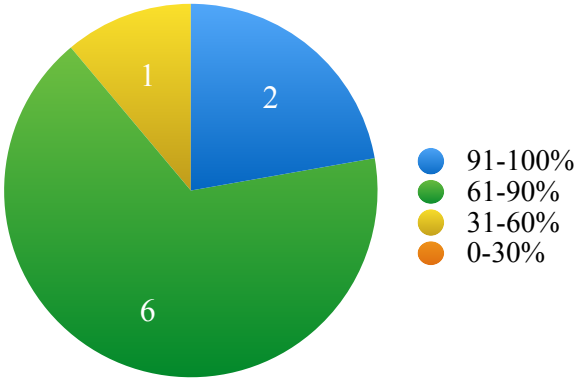
Personen som svarade annat, utvecklade sitt svar med följande:

1. *”Både ja och nej”*

6.3.5 Fråga 5

”Hur många QR-koder hittade du enkelt? (uppskatta)”

De olika svarsalternativen och fördelningen mellan dessa återfinns i figur 6.6.

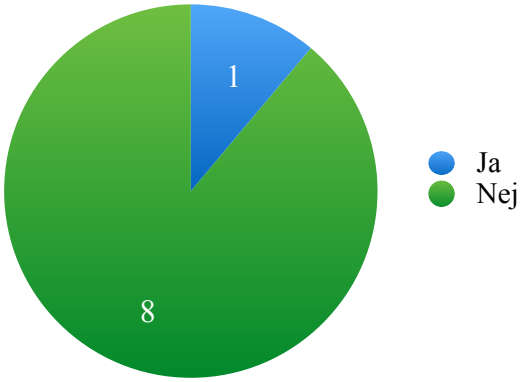


Figur 6.6 Redovisning av svar på fråga 5 ur formulär för applikation

6.3.6 Fråga 6

”Kände du dig överväldigad/belastad av den funktionalitet applikationen innehöll?”

De olika svarsalternativen och fördelningen mellan dessa återfinns i figur 6.7.



Figur 6.7 Redovisning av svar på fråga 6 ur formulär för applikation

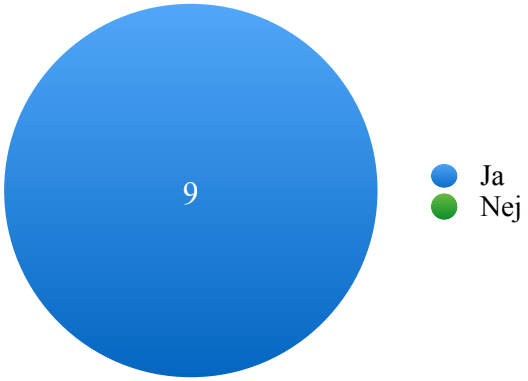
Vissa personer valde att utveckla sitt svar med följande:

1. *”Applikationen var långsam men häftig”*
2. *”Menyerna var lättöverskådliga”*
3. *”Applikationen var enkel att förstå, samt smidig att använda”*

6.3.7 Fråga 7

”Kändes applikationen lättanvänd?”

De olika svarsalternativen och fördelningen mellan dessa återfinns i figur 6.8.



Figur 6.8 Redovisning av svar på fråga 7 ur formulär för applikation

Vissa personer valde att utveckla sitt svar med följande:

1. *”Applikationen berättade vad man skulle göra”*
2. *”Tappade internetuppkoppling någon gång”*
3. *”Lätt och trevlig, även om det var mitt första försök”*

6.3.8 Fråga 8

”Om du kunde ändra en sak med applikationen, vad skulle det vara?”

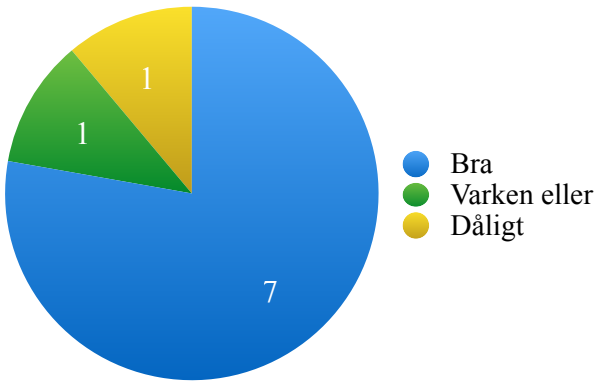
Alla deltagande svarade på denna fråga, de svar som erhölls var följande:

1. *”Språket”*
2. *”Ta bort texten ur applikationen eller ge den en mörkare bakgrund, blev svårt att läsa”*
3. *”Det kan vara överflödigt med text om språket som talas är modersmålet”*
4. *”Större text/textstorlek”*
5. *”Solen störde”*
6. *”En mer vänligt låtande röst möjligtvis”*
7. *”Större skärm”*
8. *”Snabbare val om man ska fortsätta eller få mer information”*

6.3.9 Fråga 9

”Hur kände du för att kunna läsa texten som presenterades samtidigt som den blev uppläst?”

De olika svarsalternativen och fördelningen mellan dessa återfinns i figur 6.9.



Figur 6.9 Redovisning av svar på fråga 9 ur formulär för applikation

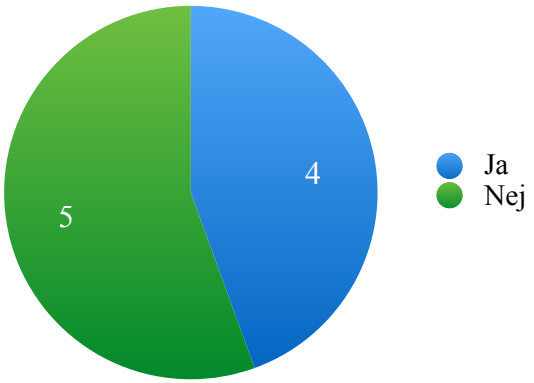
Vissa personer valde att utveckla sitt svar med följande:

1. *”Fokuseringen blev delad mellan läsande och lyssnande”*
2. *”Bra om det är högljutt runt omkring så att man kan läsa texten”*

6.3.10 Fråga 10

”Var det en bra idé att låta applikationen läsa upp alla alternativ du kunde genomföra varje gång du skannade en QR-kod?”

De olika svarsalternativen och fördelningen mellan dessa återfinns i figur 6.10.



Figur 6.10 Redovisning av svar på fråga 10 ur formulär för applikation

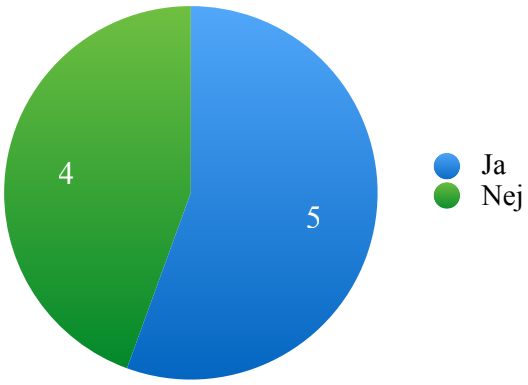
Vissa personer valde att utveckla sitt svar med följande:

1. *”Skulle vara lättare utan QR-koder, bara olika markeringar som talade om att du skulle trycka på glasögonen”*
2. *”Irriterande”*
3. *”Jag vill kunna avbryta henne”*
4. *”Tog lång tid och man väntade mycket. Man lärde sig fort att använda appen.”*

6.3.11 Fråga 11

”Stötte du på några problem med glasögonen under rundvandringen?”

De olika svarsalternativen och fördelningen mellan dessa återfinns i figur 6.11.



Figur 6.11 Redovisning av svar på fråga 11 ur formulär för applikation

Vissa personer valde att utveckla sitt svar med följande:

1. *”Högtalaren fungerade inte perfekt”*
2. *”Applikationen läste texten långsamt, irriterande att den sa samma sak om och om igen”*
3. *”Tappade internetanslutningen, tog tid att ställa in skärmen”*
4. *”Skärmen tonade bort ofta”*
5. *”Inga problem, glasögonen stängde av sig men det löste sig fort”*

7 Diskussion

I detta kapitel diskuteras resultatet projektet har åstadkommit. Inledningsvis kommer resultaten av formuläret för rundvandringen att diskuteras. Efter detta kommer formuläret för Google Glass att diskuteras för att slutligen diskutera slutprodukten projektet genererat.

7.1 Rundvandringarna

Sammanställningen av formuläret visade att informationen som gavs av en mänsklig guide är likvärdig eller bättre än informationen som presenterades av Google Glass.

Författarna tror att anledningen till att en mänsklig guide får likvärdigt eller högre resultat i samtliga frågor kan bero på att en människa kan anpassa innehållet som presenteras för de som lyssnar. En mänsklig guide kan också besvara frågor som ställs, samtidigt som den kan upplysa om ytterligare information om det anses lämpligt.

Att resultatet för de två sista frågorna för Google Glass var betydligt sämre än de tidigare frågorna kan vara på grund av den monotona röst som glasögonen har. Användaren fick också alltid information om hur de skulle agera för att ta sig vidare, detta kan tänkas upplevas som irriterande. I slutet av rundturen hade glasögonen en tendens att bli överhettade, vilket ledde till att rösten som läste upp texten började hacka och blev svår att förstå.

Applikationen försökte utformas så att det var minimal belastning på hårdvaran, det visade sig senare att de optimeringsförsök författarna försökte utföra inte hade någon påverkan på applikationen.

Det som kan ändras med applikationen för att ge ett bättre resultat är att användaren kan stänga av användningsinstruktionerna. Eftersom rösten då skulle upprepa sig mindre, leder det till att den inte blir ett lika stort irritationsmoment, detta i sin tur kanske skulle leda till att informationen som glasögonen försökte förmedla blev lättare att ta till sig. Att rösten som applikationen hade började hacka, kan endast åtgärdas med bättre hårdvara som inte lika lätt blir överhettad.

Den slutsats som får stöd från undersökningen är att människor upptar mer information när de har en mänsklig guide som de kan ställa frågor till.

7.2 Glasögonen

Google Glass är i skrivande stund en produkt som ej är etablerad på marknaden. Detta bidrar till att gemene man kan uppleva den som främmande då de ej provat tekniken. Att Google Glass har en kamera kan vara en av de största sociala faktorerna som verkar

avskräckande eftersom folk de möter upplever situationen som obekväm med en kamera riktad mot ansiktet. Om kameran hade tagits bort hade den sociala ”skrämselfaktorn” minskat avsevärt, men detta hade även påverkat användningsområdet för Google Glass avsevärt. En annan viktig faktor till att de ej känns bekväma att använda i offentligheten är det höga priset, en stöld eller olycka skulle kunna ha stor finansiell påverkan hos ägaren.

7.3 Hemsida och Applikation

Systemet som har arbetats fram har många aspekter som kan ha utförts på andra sätt. Så som en annan design på databas till att ha använt en annan lösning än QR-koder. För förbättringar på systemet i sin helhet ser inte författarna några större utrymmen för utveckling. En aspekt skulle kunna vara att utveckla en applikation till mobiltelefon också. Detta hade skapat en större målgrupp kring användarna, men eftersom projektet var avsett att lösa problemet med en applikation för Google Glass så gjordes endast en sådan. Databasdesign är något som kan lösas på flera olika sätt. Att förändra relationen mellan Area och Media så att relationen innefattar beskrivningen av bilden skulle resultera i att samma bild kan länkas till flera areor med olika beskrivningar, detta skulle potentiellt kunna spara plats på databasen ifall samma bild används flera gånger. Detta skulle även innebära att relationstabellen skulle behöva skapas och ifall samma bild aldrig används skulle denna tabell endast länka olika areor med olika bilder och därför ta upp onödig plats. Hur den nya designen ska lösas rent implementationsmässigt går att diskuteras, till exempel skulle administratören först kunna få välja bland de existerande bilderna eller ladda upp en egen vid skapandet av Area. Systemet skulle också själv göra en kontroll om bilden redan existerar genom att jämföra bildens bytes, även detta skulle vara onödigt påfrestande på systemet vid varje uppladdning. I dagsläget saknas det en gräns för hur stora bilder som får laddas upp eller vad för filtyp som får laddas upp, detta kan ställa till problem ifall en administratör väljer fel fil vid uppladdning. Ifall många administratörer skulle laddat upp stora filer skulle detta kunna fylla databasen och göra att applikationen blir oanvändbar. Ytterligare ett sätt att förbättra bildhanteringen är att skala om bilder som skickas från servern för att spara minne på glasögonen. Detta skulle dock försvåra ifall en mobilapplikation skulle utvecklats eftersom bilder i full storlek skulle vara att föredra.

7.4 Formulär för applikation

Frågorna som ställdes om vad användarna tycke om delar i applikationen kommer att diskuteras enskilt och det kommer ges förklaringar och förslag på vad som kan förbättras.

7.4.1 Fråga 1

”Hur kändes det att ha på sig Google Glass?”

Majoriteten av användarna kände sig antingen tuffa, neutrala eller inte speciella med glasögonen. Dock upplevde nästan hälften av användarna att glasögonen var fjantiga,

vilket kan vara en bidragande faktor till att de inte upptar lika mycket information när de får rundvandringen.

Eftersom Sigmas kontor är en arbetsplats som flera personer vistas i dagligen, kan det uppfattas som att besökaren inkräktar på personalens integritet, då glasögonen kan upplevas som en kamera monterad på huvudet. Röstkommandon som användes kan ha bidragit till att besökaren upplevt att de stört omgivningen.

Den sociala aspekten med att bruka smarta glasögon är ännu inte etablerad hos allmänheten vilket sannolikt kommer att ändras i framtiden, det i sin tur kanske också förändrar hur besökaren upplever glasögonen.

Eftersom Google ansvarar för designen av hårdvaran finns det väldigt lite rum för utveckling.

7.4.2 Fråga 2

”Hur mycket av informationen förstod du? (uppskatta)”

De personer som fick rundvandringen förstod all eller nästan all information som presenterades. En bidragande faktor kan ha varit att användarna både visuellt kunde läsa informationen och fick den uppläst.

Eftersom användarna upplevde att de förstod informationen som presenterades behöver inte någon förändring i applikationen göras angående förmedlingen av information.

Dock kan detta resultatet upplevas som märkligt eftersom användarna inte tillgodogjorde sig all den information som presenterades och svarade sämre på de två sista frågorna, jämfört med gruppen som hade en mänsklig guide (se figur 6.1).

7.4.3 Fråga 3

”Föredrog du att använda röst eller touch kommandon?”

Fördelningen mellan de olika svarsalternativen var spridd, men majoriteten tyckte att det bästa sättet var att använda touch. En av anledningarna kan vara att användaren upplevde att de störde omgivningen genom att ge röstkommandon.

Röstkommandon uppfattades som bra för att det kändes naturligt att navigera i glasögonen med hjälp av den typen av kommandon. Glasögonen ger också kontinuerligt information till användaren om vilka val som var möjliga genom användning av rösten.

Den slutsats som kan dras av detta är att både röst- och touch- kommandon är bra.

7.4.4 Fråga 4

”Föredrar du guidning med smarta glasögon framför en människa?”

Majoriteten som använde glasögonen skulle ha föredragit en mänsklig guide. Detta kan bero på att det vid mänsklig interaktion finns möjlighet att ställa frågor. Guiden har också möjlighet att bättre utforma rundturen kring besökarens intressen.

Vissa användare kände sig fjantiga med glasögonen, vilket kan vara en bidragande faktor till att de skulle föredra en mänsklig guide, se avsnitt 7.2.1. När användaren går runt med en guide legitimeras även att de stör omgivningen.

Slutsatsen som kan dras är att det är svårt att ersätta en människa, oavsett hur användarvänlig applikationen är. Även om tekniken är i ett tidigt skede kan detta resultat ses som lyckat för en den första implementeringen.

7.4.5 Fråga 5

”Hur många QR-koder hittade du lätt? (uppskatta)”

De flesta användare kände att de hittade de flesta eller alla QR-koder enkelt, vilket är ett positivt resultat. Eftersom rundvandringen utfördes på en arbetsplats var det möjligt att ge korta informativa beskrivningar som möjliggjorde att användaren enkelt kunde hitta nästa kod.

Slutsatsen är att beskrivningarna till nästa QR-koder kan göras tydligare och kodernas placering kan göras mer uppenbara för att underlätta navigeringen i kontorsmiljön.

7.4.6 Fråga 6

”Kände du dig överväldigad/belastad av den funktionalitet som applikationen innehöll?”

En stor majoritet av användarna upplevde att applikationen ej var överväldigande/ belastande vilket kan anses som positivt för en förstagångsanvändare med ny teknik.

Eftersom applikationen ständigt talade om för användaren vilka alternativ som fanns och dessa successivt minskade blev applikationen enkel och lätt att förstå, även om det var fösta gången smarta glasögon användes. Under utvecklingen av applikationen låg fokus på att göra den lättanvänd eftersom det med stor sannolikhet var första gången besökaren använde smarta glasögon. De menyer som finns i applikationen designades för att användaren alltid skulle kunna komma vidare i applikationen, se appendix A.

En användare valde att utveckla sitt svar med *”Applikationen var långsam men häftig”*, det som kan förändras är att optimera applikationen eller använda mer avancerad hårdvara. Eftersom Google Glass inte har några uppgraderingsmöjligheter för hårvara ligger det utanför projektets ramar.

7.4.7 Fråga 7

”Kändes applikationen lättanvänd?”

Samtliga av deltagarna svarade att applikationen var lättanvänd, vissa användare gav även egna kommentarer som även dessa tydde på att applikationen var lättanvänd.

Stort fokus har legat kring att göra applikationen lättanvänd för en förstagångsanvändare. Slutsatsen är att applikationen är lättanvänd med de metoder och designval som använts för implementeringen.

7.4.8 Fråga 8

”Om du kunde ändra en sak med applikationen, vad skulle det vara?”

De svar som avsåg att språket i applikationen skulle ändras tolkades som att applikationen borde vara på svenska. Det optimala för applikationen är att all information presenteras på användarens modersmål men som nämnts i projektets avgränsningar ligger en sådan funktion utanför projektets ramar.

Synpunkter fanns på textens utformning och synlighet och det finns ett visst samtycke från författarna om att texten kan förbättras, men då denna funktionalitet skulle ta för lång tid att implementera fanns den med i projektets avgränsningar.

En person ansåg att valen skulle presenteras snabbare, svaret antogs handla om att det kändes onödigt att applikationen informerade användaren om de alternativ de hade vid varje presenterad arbetsyta. Detta har diskuterats mer genomgående i avsnitt 7.1.

Slutsatsen som kan dras angående frågan är att vissa av de funktioner som efterfrågades, under ett tidigt skede i projektet ansågs vara viktiga men skulle ta för lång tid att implementera. En förbättring som skulle kunna göras är att de val som läses upp för användaren går att stänga av.

7.4.9 Fråga 9

”Hur kände du för att kunna läsa texten som presenteras samtidigt som den blev uppläst?”

Majoriteten av de tillfrågade såg det som bra att kunna läsa texten samtidigt som den blev uppläst. En användare tillade också att det var bra att kunna läsa texten om det var högljutt runtomkring.

Den person som tyckte det var en dålig idé att använda sig av text och röst motiverade sitt svar med följande, *”fokuseringen blev delad mellan läsande och lyssnade”*. Att fokuseringen blev delad kan bero på att glasögonen inte var rätt inställda för individen som använde glasögonen och därför inte såg hela texten som presenterades. Det som kan förändras är den information som ges till besökaren innan denne ska starta rundvandringen, för att upplysa om de inställningar som kan göras.

Slutsatsen som kan dras är att majoriteten av användarna uppskattade att få välja vilket sätt de konsumerade den information som presenterades.

7.4.10 Fråga 10

”Var det en bra idé att låta applikationen läsa upp alla alternativ du kunde genomföra varje gång du skannade en QR-kod?”

Fördelningen mellan de två svarsalternativen var mycket lika, fem personer tyckte det var dåligt och fyra personer tyckte det var bra. De personer som valde att utveckla sina svar visade en tendens att bli irriterade och otåliga när de presenterades samma information flera gånger.

Nämnvärt är att nästan hälften av användarna tyckte att det var bra att informationen presenterades varje gång, vilket kan tyda på att de kände sig trygga då de alltid visste vad nästa steg var.

Applikationen skulle kunna vidareutvecklas för att implementera en funktionalitet som möjliggör att användaren blir tillfrågad om denne vill fortsätta få instruktionerna uppläst efter några skannade koder.

Slutsatsen som kan dras är att för personer som vill kunna stänga av uppläsningen blir den irriterande och för personer som inte känner sig bekväma med den nya tekniken är den nödvändigt.

7.4.11 Fråga 11

”Stötte du på några problem med glasögonen under rundvandringen?”

Fler än hälften av användarna säger att de hade problem med glasögonen under rundvandringen. De personerna som valde att utveckla sina svar hade ofta problem med hårdvaran, som projektet inte var ämnat att lösa.

8. Slutsats

Detta kapitel ämnar att besvara rapportens akademiska frågeställning samt att klargöra hur de blev besvarade. Den första frågan som klargörs är hur ett system med smarta glasögon kan utformas för guidning i kontorslandskap. Därefter kommer frågan om vad som krävs av applikationen för att en förstagångsanvändare ska kunna förstå hur hård- och mjukvara används att besvaras. Slutligen kommer frågan om hur en rundvandring med smarta glasögon jämför sig med en rundvandring som har mänsklig guide att besvaras.

8.1 Utformningen av systemet

Utformningen av systemet kan se ut på många olika sätt, men klart är att en serverlösning bör implementeras eftersom glasögonen har begränsat lagringsutrymme och beräkningskraft. Systemet behöver även ett tillvägagångssätt för att berika de olika arbetsytor som skall presenteras under guideningen och en lättanvänd applikation.

Att applikationen måste vara lättanvänd beror på att smarta glasögon i skrivande stund inte är en väletablerad produktkategori, vilket leder till att det med stor sannolikhet är den första gången besökaren/användaren använder en produkt ur kategorin.

Systemet måste också vara utformat på ett sådant sätt att anställda på ett företag enkelt kan förnya rundturen, såsom skapa nya ytor, uppdatera den lagrade informationen och radera gammalt innehåll.

8.2 Vad krävs av applikationen

Det som krävs av en applikation till smarta glasögon för att en förstagångsanvändare ska klara av att använda dem på egen hand är ett mycket lättförstått användargränssnitt. Applikationen bör även ha minimalt med menyer som håller en enhetlig layout.

Enligt undersökningen som gjordes bidrog även användarens egna val av hur innehållet skulle konsumeras och den kontinuerliga informationen, till att applikationen var lättanvänd.

De metoder som implementerades i applikationens användargränssnitt, se avsnitt 5.2.3, har enligt undersökningen visats underlätta hur en användare upplever mjukvaran.

8.3 Hur jämför sig rundvandringarna

Enligt den undersökning som gjordes under projektet uppfattade besökare/användare av Google Glass mindre information än vad de med en mänsklig guide gjorde. En bidragande faktor till att en mänsklig guide var bättre antogs vara att frågor kunde ställas under rundturen. En annan bidragande faktor kan vara att det var första gången

besökaren/användaren använde sig av smarta glasögon, vilket ledde till att större fokus lades på hård- och mjukvara, medan mindre fokus låg på rundvandringen.

8.4 Framtida arbete

I applikationen finns en rad förbättringar att göra. En förbättring skulle kunna vara att man får frågan om man vill stänga av uppläsningen av alternativ vid varje punkt. Fördelen med detta skulle vara att användare skulle störa sig mindre på rösten och applikationen skulle inte överhettas lika enkelt eftersom röstsuppläsning tar mycket av processorkraften. Nackdelen med detta är att om en ny person skall gå rundvandringen måste denne aktivt aktivera rösten eller starta om applikationen. Detta kan leda till att användare som ej aktiverar rösten missar viktig information. Som applikationen ser ut i skrivande stund så behöver den ej startas om, det är bara att fortsätta köra applikationen varje rundtur.

Referenser

1. Personlig kommunikation, Linus Vidberg, mars 2015, Sigma.

2. Translate API Client Library for Java, Google 2015, https://developers.google.com/api-client-library/java/apis/translate/v2 (Acc: 2015-05-07)

3. Scrum, https://www.scrum.org/ (Acc: 2015-05-11)

4. Active Server Pages, Microsoft 2015, https://msdn.microsoft.com/en-us/library/aa286483.aspx (Acc: 2015-05-12)

5. Razor Intro, W3Schools 2015, http://www.w3schools.com/aspnet/razor_intro.asp (Acc : 2015-05-13)

6. Introduction to ASP.NET Web Programming Using the Razor Syntax (C#), Microsoft 2014, http://www.asp.net/web-pages/overview/getting-started/introducing-razor-syntax-%28c%29 (Acc: 2015-05-13)

7. What is MS SQL?, Host Shopper 2008, http://www.host-shopper.com/what-is-ms-sql.html (Acc: 2015-05-13)

8. Simple Example of MVC (Model View Controller) Design Pattern for Abstraction, rj45, 2008-04-08, Code Project, http://www.codeproject.com/Articles/25057/Simple-Example-of-MVC-Model-View-Controller-Design (Acc: 2015-05-27)

9. Learn About ASP.NET, Microsoft 2015, http://www.asp.net/mvc (Acc: 2015-05-27)

10. Glass Development Kit, Google 2015, https://developers.google.com/glass/develop/gdk/index (Acc: 2015-05-22)

11. Platform overview, Google 2015, https://developers.google.com/glass/develop/overview (Acc: 2015-05-22)

12. Rickard Edman, Christoffer Wernberger, Aftonbladet, först med att knäcka koden?, Göteborgs Universitet 2007, https://gupea.ub.gu.se/bitstream/2077/10002/1/640.pdf (Acc: 2015-05-13)

13. GPS, Nationalencyklopedin 2015, http://www.ne.se/uppslagsverk/encyklopedi/l%C3%A5ng/gps (2015-05-14)

14. United States Updates Global Positioning System Technology, Cheryl Pellerin IIPDigital 2007, http://iipdigital.usembassy.gov/st/english/article/

2006/02/20060203125928lcnirellep0.5061609.html#axzz3a6PSkRW0 (Acc: 2015-05-14)

15. GPS Accuracy, U.S Air Force 2014, http://www.gps.gov/systems/gps/performance/accuracy/ (Acc: 2015-05-14)

16. Who used scrum and why?, Scrum Alliance, https://www.scrumalliance.org/why-scrum/who-uses-scrum (Acc: 2015-05-11)

17. Trello Inc 2015, https://trello.com/ (Acc: 2015-05-11)

18. All About the Waterfall Model, Waterfall Model 2015, http://www.waterfall-model.com/ (Acc 2015-05-18)

19. XLC Process Overview, Centers for Medicare & Medicaid Services 2014, http://www.cms.gov/Research-Statistics-Data-and-Systems/CMS-Information-Technology/XLC/ (Acc 2015-05-18)

20. Lakeworks 2009, http://commons.wikimedia.org/wiki/File:Scrum_process.svg (Acc: 2015-05-18)

21. Visual Studio, Microsoft 2015, https://www.visualstudio.com/ (Acc: 2015-05-18)

22. Android, Google 2015, https://developer.android.com/sdk/index.html (Acc: 2015-05-18)

23. Google Glass, Google 2015, https://developers.google.com/glass/design/ (Acc 2015-05-18)

24. Saving Data in SQL Databases, Android, http://developer.android.com/training/basics/data-storage/databases.html (Acc: 2015-05-27)

25. Tech Specs, Google 2015, https://support.google.com/glass/answer/3064128?hl=en&ref_topic=3063354 (Acc: 2015-05-24)

26. QRCode.com, Denso Wave, http://www.qrcode.com/en/ (Acc: 2015-05-27)

27. ZBar bar code reader, Jeff Brown 2010, http://zbar.sourceforge.net/ (Acc: 2015-05-27)

28. Principles, Google 2015, https://developers.google.com/glass/design/principles (Acc: 2015-05-24)

29. Audio, Google 2015, https://support.google.com/glass/answer/3311275?hl=en (Acc: 2015-05-24)

30. HowStuffWorks.com, http://s.hswstatic.com/gif/bonephones-illustration.jpg (Acc: 2015-04-20)

31. Kiger, Patrick J.. "How Bone-conducting Headphones Work" 30 April 2012. HowStuffWorks.com, http://electronics.howstuffworks.com/gadgets/audio-music/bone-conducting-headphones.htm (Acc: 2015-04-20)

32. Introduction to ASP.NET MVC, Microsoft Virtual Academy 2015, http://www.microsoftvirtualacademy.com/training-courses/introduction-to-asp-net-mvc (Acc: 2015-05-19)

33. Getting started with Entity Framework 6 Code First using MVC 5, Microsoft 2015, https://www.asp.net/mvc/overview/getting-started/getting-started-with-ef-using-mvc/creating-an-entity-framework-data-model-for-an-asp-net-mvc-application (Acc: 2015-05-12)

34. ASP.NET MVC 5 – CRUD operations with Entity Framework 6 on Visual Studio 2013, Microsoft 2014, https://code.msdn.microsoft.com/MVC5-Demo-with-Entity-c6bc81df (Acc 2015-05-12)

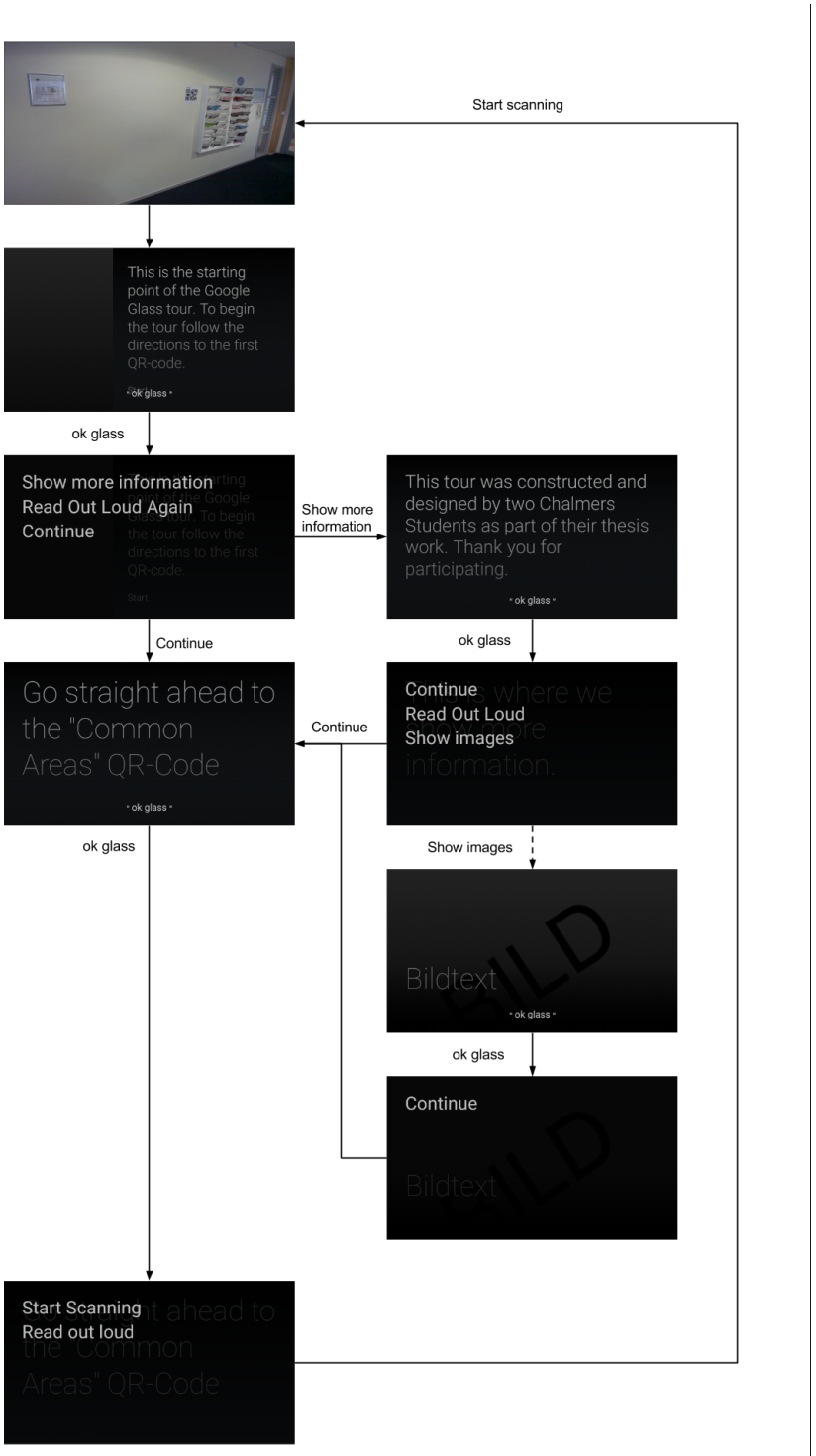
35. Get Bootsrap, Bootstrap 2015, http://getbootstrap.com/ (Acc: 2015-05-24)

36. jQuery.qrcode, Lars Jung 2015, http://larsjung.de/jquery-qrcode/ (Acc: 2015-05-24)

37. Rotativa, how to print PDF in ASP.NET MVC, Codeproject 2015, http://www.codeproject.com/Articles/335595/Rotativa-how-to-print-PDF-in-Asp-Net-MVC (Acc: 2015-05-20)

38. Bitmap, 21-05-2015, Android, http://developer.android.com/reference/android/graphics/Bitmap.html (Acc: 2015-05-27)

Appendix A



Flowchart diagram till applikationen för Google Glass

Appendix B

Survey

How did you feel while wearing the glasses?

1. Cool/Innovative
2. Normal
3. Nothing unusual
4. Silly
5. Other _____

How much of the information did you understand (Estimate)

1. Every word.
2. Some of it
3. None of it

Did you prefer using the voice commands or touch commands?

1. Voice
2. Touch
3. Only tried Voice
4. Only tried Touch

Would you prefer this way of sharing information over using a human guide?

1. Yes
2. No
3. Other _____

How many of the QR-Codes did you find easily (Estimate)?

1. 0-30%
2. 31-60%
3. 61-90%
4. 91-100%

Did you ever feel overwhelmed by all the different functions of the application?

1. Yes
2. No

Elaborate: _____
_____

Did you find the application easy to use?

1. Yes
2. No

Elaborate: _____
_____

If you could change one thing about the application what would it be?

_____
_____

How did you feel about having the text that was spoken to you readable at the same time?

1. Good
2. Neither
3. Bad

Elaborate: _____
_____

Was it a good idea to have the application tell you all the alternatives every time you scan a QR-code?

1. Yes
2. No

Elaborate: _____
_____

Did you encounter any problems with the hardware of the Google Glass?

1. Yes
2. No

Elaborate: _____
_____

Appendix C

Interrogation

What times of the day are you allowed to use the Playstation 4?

1. Only after hours
2. During lunch breaks and after hours
3. When with a customer
4. Any time you want

If you're in the sales department and your phone rings, what are you supposed to do?

1. Tell everyone around to be quiet
2. Lower your voice
3. Go away to another area

Why are there no technical equipment in the meeting room you saw?

1. It focuses on the conversation.
2. The room is to sunny

The phone booth were designed with ____ in mind?

1. Short calls
2. Telephone conferences
3. Acoustics
4. One person phone calls

Why do we have a project room?

1. If there is no room at the customers office.
2. To fully focus on a project
3. To not be disturbed or make sure to find a place.