VeloFlow

Koncept för en applikation till smartklockor som ger cyklister en grön våg i trafiken

Examensarbete inom högskoleprogrammet Datateknik

ISAK HOLMDAHL
RASMUS STANDAR

INSTITUTIONEN FÖR DATA- OCH INFORMATIONSTEKNIK
CHALMERS TEKNISKA HÖGSKOLA
Göteborg 2023
www.chalmers.se

Examensarbete 2023

VeloFlow

Koncept för en applikation till smartklockor som ger cyklister en grön våg i trafiken

ISAK HOLMDAHL
RASMUS STANDAR

GÖTEBORGS UNIVERSITET

CHALMERS

Institutionen för data- och informationsteknik
Chalmers Tekniska Högskola
Göteborgs Universitet
Göteborg 2023

VeloFlow
Koncept för en applikation till smartklockor som ger cyklister en grön våg i trafiken

ISAK HOLMDAHL
RASMUS STANDAR

Handledare: Sara Ljungblad, Institutionen för data- och informationsteknik
Tekniska handledare: Alex Darborg & Terje Engelbertsen, Knightec
Examinator: Lars Svensson, Institutionen för data- och informationsteknik

Examensarbete 2023
Institutionen för data- och informationsteknik
Chalmers Tekniska Högskola
SE-412 96 Göteborg
Telefon +46 31 772 1000

Omslagsbild: Studenterna bakom detta arbete som testar sin applikation.

Skriven i LaTeX
Göteborg 2023

VeloFlow
Koncept för en applikation till smartklockor som ger cyklister en grön våg i trafiken

ISAK HOLMDAHL
RASMUS STANDAR
Institutionen för Data- och informationsteknik
Chalmers Tekniska Högskola
Göteborgs Universitet

Sammanfattning

Cykeln är ett smidigt transportmedel som är miljövänligt och billigt att använda. Trots det är det inte tillräckligt många som väljer att ta cykeln istället för bilen. I detta examensarbete har en smartklockeapplikation, VeloFlow, utvecklats tillsammans med en molntjänst som simulerar trafikljus. VeloFlow skapades för att utforska om det går att skapa en applikation som kan ge cyklister en grön våg i trafiken, samt se hur en applikation kan kommunicera med cyklister på ett säkert och distraktionsfritt vis. Trafikljussimuleringen hjälpte till i utvärderingen utav applikationen för att se om konceptet kan fungera, och resultatet visar att en framtida version av VeloFlow som kopplas till riktiga trafikljus kan hjälpa cyklister att få en grön våg, vilket i sin tur kan få fler att välja cykeln istället för bilen. För att hjälpa till i vidare utveckling och forskning inom ämnet så sammanfattas detta arbetets resultat i sex stycken designfaktorer. Dessa designfaktorer hjälper till att visa vilka komponenter som applikationer för cyklister ska innehålla och vilka metoder som är viktiga för utvecklingen av dessa applikationer.

Nyckelord: Cykel, UCD, Göteborg, Trafikljus, Wear OS, Smartklocka, Compose, Android, Grön våg.

Förord

Vi som skriver detta examensarbete vill tacka våra handledare Terje Engelbertsen, Alex Darborg och Sara Ljungblad för det stöd de gett under arbetets gång och den vägledning som gjort detta arbete till vad det är. Stort tack till Otto Nilsson som bidrog med sin röst och inspelningsexpertis för våra ljudmeddelanden. Vi vill också tacka alla på Knightec som tagit sin tid på att lyssna på oss och vårt ständiga prat om cykling. Ett extra stort tack ska ges till alla på avdelningen Dewire i Göteborg, som agerade testpersoner och som hjälpt till med massviss av idéer för projektet. Slutligen vill vi även tacka våra kurskamrater på Chalmers som hjälpte oss ta fram användarkrav genom att testa vår prototyp i både regn, rusk och kyla.

Isak Holmdahl, Rasmus Standar, Göteborg, Maj 2023

Akronymer

Nedan listas akronymer som används genom denna rapport i alfabetisk ordning:

API	Application Programming Interface
AWS	Amazon Web Services
HTTP	Hypertext Transport Protocol
MVVM	Model View Viewmodel
UCD	User-centered design

Innehåll

Figurer

1

Inledning

1.1 Bakgrund

Andelen cykelresor i Göteborg är låg; enbart 7% av alla resor skedde med cykel år 2019, samtidigt som 43% av alla resorna skedde med bil. För att sätta siffrorna i kontext så är motsvarande siffra för andel cykelresor 26% i Malmö, 28% i Köpenhamn, 33% i Uppsala och 13% i Berlin[1]. En ökning av cykelresor skulle direkt kunna hjälpa till att uppfylla fem av sjutton av de Globala målen eller indirekt uppfylla tio av de sjutton målen[2],[3]. Göteborg hamnar dessutom på sista plats i Cykelfrämjandets undersökning kring cyklisters nöjdhet inom gruppen stora kommuner[4]. För att få Göteborg att bli attraktivare som cykelstad och för att öka andelen cyklister behöver flera satsningar utföras. Detta examensarbete strävar efter att bidra till att skapa en lösning för att få grön våg som kan hjälpa till att öka andelen cykelresor och göra cykling i Göteborg till en bättre upplevelse. En grön våg innebär att cyklisten inte behöver stanna vid trafikljus utan alltid anländer vid trafikljus när de är gröna.

I detta kapitel presenteras bakgrunden till projektet med information om fördelar med cykelresor, vilka mål som idag finns för cykling och hur dessa kan höjas, vilka parter som varit involverade, vilket syfte och mål projektet har samt vilka avgränsningar som gjort.

1.1.1 Fördelar med att ta cykeln istället för bilen

För att få bukt på den hälsofara som drabbar världens städer och då i längden negativt påverkar den sociala hållbarheten behöver mängden utsläppta avgaser från bilar minska. En studie har utförts av forskare från bland annat Stockholms universitet och Umeås universitet som utforskade effekterna av vad som hade hänt om personer i Stockholm som har en pendelsträcka på max 30 minuter med cykel valde att ta cykel istället för bil. Flera positiva effekter uppdagades, så som den ekologiska påverkan att mängden utsläpp av den hälsofarliga bil-avgasen NO_x hade minskat med 20% i Stockholms innerstad. Denna minskning hade lett till den sociala påverkan att rädda 31 personer i Stockholms innerstad från en för tidig död, och 63 stycken i Stockholms län. Tittar man endast på utsläppet av NO_x så ser man att 449 år av livslängd hade räddats årligen i Stockholm. Studien visar att detta hypotetis-

ka byte från bil till cykel hade hade varit mer effektiv sett på räddade liv än den trängselskatt som nu finns i Stockholm [5]. Ett minskat antal bilister och ökat antal cyklister hade även haft ytterligare en ekologisk påverkan angående minskandet av växthusgaser. Bilar som drivs av fossila bränslen släpper ut växthusgasen CO_2 som ökar den globala uppvärmingen [6]. Ett minskat antal bilresor hade alltså inte bara gett en positiv samhällspåverkan utan även en positiv ekologisk påverkan.

1.1.2 Oambitiösa mål för cykling i Göteborgs stad

I rapporten ”Cykelprogram för en nära storstad 2015-2025” sätter Göteborg stad flera cykelrelaterade mål för Göteborg. Ett av dessa mål var att till år 2025 ska antalet cykelresor ha tredubblats jämfört med 2015 vilket hade resulterat i att uppskattningsvis 12% av de totala resorna sker med cykel. Ytterligare ett mål är att 3 av 4 göteborgare ska tycka att Göteborg är en bra stad för cyklister. Som argument för att öka cyklingen i Göteborg nämns i projektbeskrivningen att ökad cykling har positiva effekter på folkhälsan och miljön, samt att det främjar stadslivet. Ett utav de planerade verktygen för att öka antalet cyklister är att ”Förbättra trafiksignalernas anpassning till cyklister”, som innefattar att ge cyklister högre prioritet vid trafikljus, ge cyklister en grön våg och minska antalen stopp för cyklister vid trafikljus [7]. Projektet ser tyvärr ut att ligga efter i sina mål. Under år 2021 låg ökningen av cykelresor på endast 36%. Som tidigare nämnt är målet 200% ökning till 2025. Även målet att fler göteborgare ska tycka att Göteborg är en bra stad för cyklister ligger en bra bit efter sitt mål. Det uppmätta värdet har sedan 2008 legat runt 40%, och målet här är 75% [8].

Naturskyddsföreningen kritiserar under sin ståndpunkt *Göteborg är ute och cyklar* de mål som Göteborg har satt upp för att inte vara tillräckligt ambitiösa [1]. I stället för att öka antalet cykelresor till 12% 2035 så förespråkar Naturskyddsföreningen en ökning till 20% av alla resor samtidigt som de vill att andelen bilresor ska ner till 15% av resorna i stället för Göteborgs mål på 29%. Som argument till varför Göteborg borde ha mer ambitiösa mål lyfter Naturskyddsföreningen bland annat följande fördelar: att det är platseffektivt med cykling, att det är bra för hälsan, att luftkvaliten ökar, att det blir mindre trångt på kollektivtrafik och att det minskar klimatpåverkan.

1.1.3 Knightec

Problemformuleringen för detta examensarbete skapades av företaget Knightec. Knightec såg att det miljövänliga transportmedlet cykel blir underprioriterat i trafiken, och att lösningen för att få fler människor att välja cykeln istället för bilen är att ge mer incitament till att välja cykeln. Incitament till cyklister kan ges på flera olika sätt. Två av dessa som Knightec hade som förslag på projekt är att se över möjligheten att ge cyklister högre prioritet vid trafikljus och att genom förmedling av information från trafikljus ge cyklister möjligheten att anpassa sin hastighet för att få en grön våg. I dialog med Knightec beslutades att detta examensarbete kommer att fokusera på att ge cyklister grön våg genom att utveckla en smartkloc-

keapplikation som kan förmedla information om trafikljusstatus och hur cyklisten ska anpassa sin hastighet för att komma i rätt tid.

Knightec tillhandahåller under examensarbetet arbetsplats på deras kontor i Göteborg, en smartklocka för att testa applikationen samt två handledare som erbjuder vägledning och teknisk kunskap.

1.1.4 Möjliga lösningar

I programmet från Göteborgs trafikkontor [7] är en av de föreslagna lösningarna för att öka antalet cykelresor i Göteborg en förbättring av infrastrukturen för cyklister. Programmet föreslår bland annat att pendlingscykelnätet ska byggas ut och att det ska nå vissa bestämda mål kopplat till säkerhet, framkomlighet och komfort. Problemet med en utbyggnad och förbättring av infrastruktur är de kostnader som ett sådant arbete innefattar. Det blir en kostnad inte bara monetärt men även i tid. En lösning som inte innefattar ombyggnationer och planering av hur dessa ombyggnationer ska utföras skulle resultera i en mindre kostnad. Trots detta är utbyggnad fortfarande en lösning som bör undersökas och övervägas, men detta arbete kommer att utforska en annan lösning som kan samverka med utbyggnation utav infrastrukturen.

För att få fler att välja cykel över bil behöver cykel som transportmedel höja sin status och bli mer lukrativ att använda för personer som vanligtvis tar bilen. Ett forskningsarbete kallat ”Co-riding With My eBike to Get Green Lights” utforskade hur maskin och människa kan bli tätare sammanbundna och jobba tillsammans, och potentialen deras arbete kan ha för att förbättra situationen för cyklister. I forskningsarbetet byggde forskarna en elcykel som hjälpte cyklisten att uppnå en grön våg genom att hjälpa cyklisten komma upp i 22km/h, den optimala hastigheten för att få en grön våg i den stad de utförde projektet, och sedan behålla den farten. Cyklisten undvek att åka för snabbt genom att cykeln, som fick namnet Ari, meddelade cyklisten genom ett par hörlurar att cyklisten borde sänka farten om den åker snabbare än 22km/h [9].

Den lösning som detta examensarbete kommer fokusera på är utvecklingen av en applikation för smartklockor som enkelt ska kunna användas av cyklister och indikera till dem när trafikljus håller på att slå om så att cyklisten ska veta om den bör höja eller sänka sin hastighet för att anlända vid trafikljuset när det är grönt.

1.2 Syfte

Syftet med detta projekt är att undersöka om det går att underlätta för cyklister att få bra flyt i sin cykling med hjälp av en smartklockeapplikation som ska hjälpa cyklisten att få en grön våg vid trafikljus, men framförallt att utforska vilka designfaktorer som faktiskt är viktiga när man skapar denna och liknande applikationer. Genom att cyklisten får en grön våg så blir hens cykling mer energieffektiv då det inte finns något behov av att stanna för att sedan accelerera upp i hastighet igen.

En grön våg kan även göra att det går snabbare att ta sig fram då cyklisten alltid kommer att vara i rörelse. Allt detta kan i sin tur leda till att allt fler väljer att cykla istället för att ta bilen om VeloFlow utvecklas till en fullfärdig applikation.

Det finns flera olika metoder för att få en grön våg, i detta projektet kommer fokus ligga på att undersöka om det går att få till en grön våg med hjälp av en applikation för smarta klockor som hjälper cyklisten att cykla i rätt hastighet med hjälp av data angående trafikljusens status.

VeloFlow har en primär arbetsuppgift: att förmedla instruktioner till cyklisten. Applikationen ska kunna använda sig av information från trafikljus (eller en simulering av trafikljus) för att delge cyklisten information om hur hen ska cykla för att undvika behovet att stanna vid ett trafikljus. För att det ska vara säkert för cyklisten att använda VeloFlow så måste applikationen designas så att den går att använda utan att användaren distraheras. För att avgöra vilka instruktioner som ska ges till användaren måste även VeloFlow kunna hantera användarens platsinformation och hastighet för att veta vilket avstånd som användaren befinner sig från olika trafikljus för att kunna bedöma om cyklisten ska öka eller sänka sitt tempo.

Utöver utvecklingen av applikationen VeloFlow för smarta klockor kommer projektet även innefatta utvecklingen av en molntjänst samt utveckling av ett applikationsprogrammeringsgränssnitt (API) för att kommunicera mellan molntjänsten och applikationen.

Molntjänsten ska användas för att simulera trafikljus och agera datahanterare för dessa trafikljus, samt räkna ut vilket trafikljus som användaren är på väg mot. Den ska kunna skicka ut information om hur lång tid trafikljuset som användaren är på väg mot kommer att vara rött eller grönt och var trafikljuset befinner sig. Datan som molntjänsten behöver för att avgöra vilket trafikljus som ska väljas kommer från VeloFlow och förmedlas med hjälp av API:t.

Den forskningsfråga som arbetet syftar att svara på är:

Vilka designfaktorer är viktiga när man skapar en smartklockeapplikation som underlättar för cyklister att få en grön våg?

1.3 Mål

När arbetet är avslutat så ska applikationen VeloFlow till smarta klockor med operativsystemet Wear OS ha utvecklats. Applikationen ska hjälpa till att undersöka om det går att underlätta för cyklister i Göteborg genom att informera om status på simulerade trafikljus i närheten av cyklisten. Detta innebär att VeloFlow ska veta om ett trafikljus kommer att slå över från grönt till rött snart, och tvärt om, för att veta om användaren ska anpassa sin hastighet för att få en grön våg. Ytterligare ett krav på applikationen är att den ska ha ett simpelt användargränssnitt som inte ska riskera att ta användarens uppmärksamhet ifrån trafiken. För att hålla risken för

olyckor på grund av distraktion minimal så krävs det att användaren inte behöver titta på klockan mer än absolut nödvändigt för att få ut information under tiden som användaren cyklar. När användaren inte cyklar så får skärmen användas fritt, och även när användaren står stilla får skärmen användas.

Vid arbetets slut ska även en simulering av ett system av trafikljus ha konstruerats så att applikationen går att testa.

Dessa delar ska sedan användas för att utvärdera om VeloFlow hade gått att använda för att underlätta cykling.

1.4 Avgränsningar

Arbetet innefattar att skapa en högfungerande prototyp av den föreslagna applikationen. Applikationen ska fungera i den testmiljö som kommer att utvecklas i form av simulerade trafikljus och den ska kunna ge en indikation på om en fullt utarbetad version hade gjort det mer lockande för personer i trafiken att välja cykel över bil. En avgränsning för projektet blir alltså att inte utveckla en komplett applikation som ska lanseras öppet för allmänheten, utan en applikation avsedd endast för att se om applikationen är realistisk och för att förstå dess möjligheter och begränsningar i ett tidigt skede.

Komplexiteten av de simulerade trafikljusen är endast till en nivå som räcker för att utvärdera ifall konceptet för VeloFlow är rimligt. Detta innebär att endast ett mindre antal trafikljus kommer att simuleras och att korsningar av flera trafikljus inte kommer att simuleras eftersom interaktionen mellan flera trafikljus vid samma korsning hade ökat komplexiteten utan att ge mer värde för utvärderingen av applikationens grundkoncept. Se avsnitt 4.3.1 för ytterligare diskussion angående de simulerade trafikljusens komplexitet.

Applikationen kommer endast att utvecklas för Wear OS plattformen. Detta är ett förslag från Knightec då det för frågeställningen inte krävs en applikation som fungerar på flera plattformar.

Frågeställningen för detta projekt riktar sig endast mot cyklister och att göra dem mer prioriterade i trafiken. VeloFlow kommer inte att hantera gångtrafikanter eller andra individer i trafiken. Fordon som elsparkcyklar och andra fordon som delar cykelvägar med cyklister skulle kunna vara möjliga användare. Projektet kommer trots detta avgränsa sig till cyklister då det annars hade krävts utökad testning för att inkludera fler fordon.

2

Metod

I detta kapitel förklaras metoder som kommer att användas i projektet, och även översiktliga förklaringar om hur arbetet kommer att genomföras presenteras.

2.1 Användarcentrerad design

Användarcentrerad design, eller den mer kända engelska benämningen user-centered design (UCD), är en designfilosofi där användaren står i fokus. Det är den tänkta användaren för ett system som ska vara grunden för de designval som görs, och denna användaren behöver därför förstås till fullo. Användarens olika behov, aspirationer samt svagheter ska stå i centrum för att se till att den utvecklade produkten ska ha så stor användarbarhet som möjligt. UCD kräver därför att frekvent testning och utvärdering görs för att se till att designprocessen inte svävar bort från användarens faktiska krav. Målet är inte att göra en design som verkar rimlig utifrån den som designar, då denna inte alltid delar krav med användaren [10].

Dålig implementation utav UCD kan resultera i produkter som är både hämmande och farliga. Om produkten inte är gjord med användaren i fokus så finns det inte en garanti för att produkten kommer att förstås utav användaren [10]. I fallet med en applikation för cyklister så kan en icke användarcentrisk design resultera i en osäker produkt som kan medföra allvarliga fysiska skador för användaren i händelsen av cykelolyckor.

Följande underavsnitt presenterar det främsta verktyget för att lyckas med UCD: användardatainsamling.

2.1.1 Insamling utav användardata

För att få förståelse för användaren och dess färdigheter och krav så behöver användarkrav insamlas. Denna data kan samlas in i användarstudier, och dessa studier kan utföras med hjälp av flera olika verktyg. De olika verktygen har olika styrkor och svagheter och kan därför komplettera varandra på ett sätt där en kombination av tillvägagångsätt kan vara det bästa alternativet [11].

Här följer en förklaring utav två olika tillvägagångssätt för att samla in användardata som används i detta arbete.

2.1.1.1 Datainsamling genom intervjuer

Intervjuer är ett verktyg för att samla in information vid användartester från de personer som är med i undersökningen. En intervju är en konversation med personen där målet är att få information genom att ställa frågor som kan vara antingen kortfattade eller kräva längre svar, och den som utför intervjun får möjligheten att ställa följdfrågor eller förtydliga frågan om det uppstår oklarheter. Denna möjlighet ges inte vid informationsinsamling gjord med hjälp av enkäter, och är den stora fördelen med intervjuer [11].

Det finns olika typer av intervjuer som har olika upplägg som skiljer dem åt. Den första varianten är ostrukturerade intervjuer. Dessa är väldigt öppna på det sättet att upplägget liknar vanliga konversationer, där öppna frågor ställs och diskussioner kan startas om det finns intresse hos den intervjuade. I en ostrukturerad intervju är det både den som håller i intervjun och den som intervjuas som kan styra konversationen åt det håll denna vill. Ostrukturerade intervjuer präglas av att vara utforskande, men detta betyder inte att intervjun ska hållas utan ett generellt upplägg för vilka frågor som ska ställas och utgå ifrån. Slutligen så genererar ostrukturerade intervjuer en signifikant mängd data som kan ta lång tid att sammanfatta, särkilt eftersom två intervjuer sällan har samma upplägg och flöde. Samtidigt så kan denna stora mängd information ge intressanta slutsatser [11].

Strukturerade intervjuer är den andra varianten av intervju, där samma frågor ställs i varje intervju och i samma ordning varje gång. Dessa frågor är oftast slutna, alltså av den sort där svaren som kan ges är från en fördefinierad samling av svar. En strukturerad intervju är därför lämplig när det är säkert att svar oftast kommer falla inom denna samling av fördefinierade svar. Alternativet att svara något utanför samlingen av svar kan ges men frågorna bör vara formulerade på det sättet att detta alternativ ska väljas så få gånger som möjligt [11].

I intervjuerna för de användartest som kommer att utföras för detta arbete kommer den strukturerade varianten att användas, då dessa inte tar lång tid att utföra och kravet på att ställa följdfrågor inte finns för de datainsamlingar som kommer att utföras. Det gör även den insamlade datan enklare att sammanfatta och analysera.

2.1.1.2 Datainsamling genom observationer

Observation innebär att användarna i studien kommer att observeras när de utför ett set av fördefinierade handlingar, i denna användarstudien relaterade till interaktion med en smartklocka under tiden som användaren cyklar. Datan som blir insamlad från observation kan hjälpa till att fylla i hålen som kan skapas när personer själva får svara på frågor i en intervju, och kan även hjälpa att ge en mer sanningsenlig bild av hur testpersonerna beter sig. Testpersonerna kan nämligen svara på frågor i en intervju på ett sätt som skiljer sig från hur det faktiskt är i verkligenheten, som

till exempel hur de utför en specifik aktivitet eller hur väl de kan utföra den aktiviteten. Ett exempel som kan ges är observation av en testperson som använder en vattenkokare. Testpersonen skulle kunna säga efter utfört test att det var enkelt att använda vattenkokaren och att resultatet av användingen var lyckat. Observationen av testpersonen skulle kunna visa den faktiska sanningen, att testpersonen faktiskt hade visat sig vara långsam och uppenbart förvirrad under testet och kanske till och med inte hade märkt att den hade spillt mycket vatten under testet [11].

Observationer kan göras direkt i en kontrollerad miljö men även i naturlig miljö. Det går till och med att göra en indirekt observation av personer med hjälp av dagböcker. I en kontrollerad miljö skapas det ett fokus på hur exakt personen utför en handling medan i en naturlig och öppen miljö skiftas fokuset till sammanhanget av testpersonen och omgivning den befinner sig i [11].

Observation kommer att användas som komplement till den strukturerade intervjun som görs vid första datainsamlingen. Målet med observationen är att se om testpersonernas svar korrekt speglar hur de faktiskt agerar i testerna.

2.1.2 Etiska aspekter vid datainsamling

Det skapas ett etiskt ansvar vid insamling av data från privatpersoner, och det har att göra med, vilken personlig data som samlas in, hur deras personliga data hanteras och hur den lagras [12]. För att undvika att personlig data hanteras inkorrekt och med etiska brott så kommer den data som samlas in att anonymiseras. Testpersoner kommer att få ett anonymt id i form av ett nummer som gör det möjligt att koppla intervjusvar med tillhörande observationer, men samtidigt omöjligt att identifiera individens personinformation.

Det finns även ett etiskt ansvar att inte sätta personerna i testerna i någon sorts fara som kan uppkomma vid tester där cykling utförs. Om olyckor skulle uppstå på grund av för stora distraktioner under cyklingen så är det de testansvariga som behöver ta ansvar.

En sista etisk aspekt som behöver tas i åsyn är spridningen av testpersoner. Ålder, kön, socioekonomisk bakgrund är variabler som kan påverka hur testerpersoner svarar i en undersökning och skillnaderna mellan grupper kan vara oväntad. På grund av denna okunskap om hur grupperna kan skilja sig är det viktigt att göra blandningen av bakgrunder så spridd som möjligt för att kunna få alla synvinklar. I detta projekt spelar även cykelvana in som en viktig variabel. Att exkludera en grupp från testerna skulle kunna skapa en lösning som inte alls skulle fungera för den gruppen. Det blir därför etiskt viktigt att vara inkluderande.

2.2 Designtaktik för system- och användarinteraktion

I det tidigare nämnda arbetet ”Co-riding With My eBike to Get Green Lights”[9], som utforskade interaktionen mellan ett tekniskt system och cyklister för att upp-

nå en grön våg, så presenterades sex stycken designtaktiker. Den första taktiken: ”Contextual Cues to Facilitate Skill Integration”, förklarar att för att system och användare ska kunna samarbeta så krävs det att båda parters färdigheter integreras. Båda parter delar ett mål och för att detta mål ska nås så behöver de kommunicera. I denna designtaktik finns det fyra punkter som bör tas i åtanke för att uppnå färdighetsintegrering.

En av de fyra punkterna förklarar att system ska använda sig av ”kortfattade kontextuella ledtrådar” för att förmedla information på ett enkelt sätt och på så sätt göra användandet av systemet enklare. För ”Co-riding With My eBike to Get Green Lights”[9] innebar detta att spela upp korta ljudmeddelanden till användaren genom ett par benledande hörlurar. Detta arbete kommer fortsätta fortsätta att utforska skapandet av kontextuella ledtrådar för cyklister på nya sätt genom att utnyttja de funktioner som smartklockor kan erbjuda, såsom vibrationer.

Ytterligare en punkt beskriver hur användaren ska förstå hur dess handlingar påverkar samarbetet mellan denne och systemet. För att främja färdighetsintegrering så ska systemet reglera hur mycket det tillför till processen utifrån hur användarens insats till användningen ser ut, och på det sättet får användaren bättre förståelse för relationen. Med bättre förståelse över relationen så kan användaren bättre förstå hur den kan förändra sitt bidrag under användning [9].

2.3 Agilt arbetsflöde utifrån ramverket Scrum

För att strukturera projektet så kommer arbetet följa det agila ramverket Scrum. Inom Scrum delas arbetet upp i kortare perioder kallade sprinter av förutbestämd längd där varje sprint inleds med en sprintplanering. Under sprintplaneringen sätts ett mål för den uppkommande sprinten och sedan planeras uppgifter för att kunna uppnå sprintmålet. Löpande under sprinten sker dagliga kortare möten för att stämma av tre frågor; ”Vad har jag gjort sen senaste mötet?”, ”Vad planerar jag att göra idag?” och ”Har jag något jag behöver hjälp med?”. I slutet av varje sprint sker en demonstration och ett retrospektiv. Under demonstrationen presenteras de resultat som har presterats under senaste sprinten, men också vad som var planerat men inte hann slutföras. Som uppföljning till demonstrationen kommer ett retrospektiv som syftar till att samla in de lärdomar som kommer från senaste sprintens arbete och använda dessa för att utveckla arbetet i framtiden [13].

Inom Scrum definieras tre primära roller: produktägaren, scrum-master och utvecklingsteamet. Produktägaren har i uppgift att maximera värdet av utvecklingsteamets arbete, detta sker genom att prioritera vilka uppgifter som ska utföras varje sprint. Scrum-mastern ser till att Scrum efterföljs och är den som leder sprintplaneringen, retrospekten och den ständiga utvecklingen utav teamet. Utvecklingsteamet är den grupp av personer som utför själva uppgifterna och utvecklar produkten eller tjänsten [13].

Under detta examensarbetet kommer varje sprint att vara en vecka lång och löpa

från fredag till fredag. De dagliga mötena kommer att hållas regelbundet i så stor mån som möjligt och sprintplanering, demonstration och retrospekt kommer alla att hållas på fredagar. Utvecklingsteamet kommer att bestå av de två examensarbetarna och produktägare för projektet kommer vara handledarna på Knightec. Då utvecklingsteamet enbart består av två personer så kommer det inte finnas en specifik scrum-master, utan ansvaret som scrum-mastern brukar ha hamnar i stället på hela utvecklingsteamet.

3

Teknisk Bakgrund

Detta kapitel introducerar de tekniska verktyg som använts under projektets gång.

3.1 Utveckling för Android

VeloFlow är en Android-applikation skapad med hjälp av ett antal verktyg, så som ramverk och designmönster. I detta kapitel förklaras de verktyg som använts.

3.1.1 Wear OS

Wear OS är en version av Android för bärbara enheter, främst smarta klockor. Operativsystemet är utvecklat av Google och är likt Android för mobiltelefoner, men skiljer sig på vissa punkter så som utformning av användargränssnitt och navigering inom applikationer [14].

3.1.2 Jetpack Compose

Jetpack Compose är det av Android rekommenderade verktyget för att bygga användargränssnitt för Android [15]. Fördelar med Compose är att det krävs mindre mängd kod för samma funktionalitet i jämförelse med tidigare verktyg och att applikationerna är effektivare då applikationer som använder Compose bara om-renderar funktioner där in-data har ändrats [15],[16].

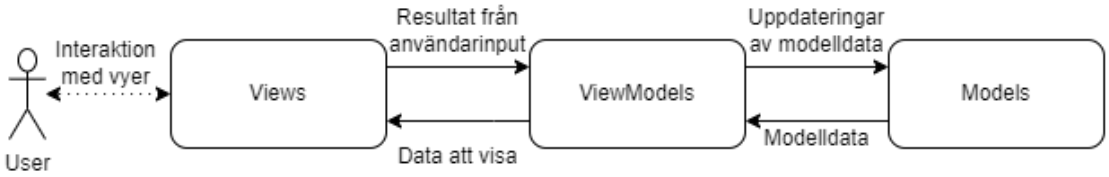
För att skapa grafiska komponenter, Composables, så räcker det att skapa en funktion i Kotlin och ge den annoteringen @Composable. En Composable renderas vid start av applikationen, och när indata som parameter till funktionen ändras. En Composable kan dessutom innehålla andra Composables, vilket möljliggör design av komplicerade, men samtidigt modulära användargränssnitt.

Jetpack Compose använder sig av deklarativ programmering och inte imperativ programmering [17]. Detta innebär att fokus är på vad koden ska göra och inte hur. Inom imperativ programmering som är den vanligaste paradigmen så beskriver programmeraren explicit vilka instruktioner och i vilken följd dessa instruktioner ska köras [18]. I imperativ programmering beskriver programmeraren i stället bara

vad den vill att programmet ska göra och överlämnar till kompilatorn att bestämma vilket kontrollflöde som ska användas.

3.1.3 Designmönstret MVVM

För att få en bra struktur som är enkel att bygga ut och göra ändringar i, och särkilt enkel att felsöka i, så har designmönstret MVVM använts i detta projekt. MVVM, som står för Model View Viewmodel, är ett designmönster som kan användas i utvecklingen av mobila applikationer. Mönstret delar upp programmet och gör det tydligt vad varje del har för arbetsuppgift, och på det sättet blir det tydligt vart förändringar ska göras eller var fel kan uppstå. I figur 3.1 syns struktur för MVVM och vilken information som ska skickas mellan de olika delarna. Vyerna i utveckling för Jetpack Compose blir *Composables*.



Figur 3.1: Diagram över datautbyte inom MVVM

3.1.4 Android Studio

Utvecklingsmiljön som använts under detta projekt för applikationsutveckling har varit Android studio, som är den officiella arbetsmiljön för utveckling av applikationer för Android-enheter. Android Studio är byggt på IntelliJ IDEA men erbjuder ytterligare funktionalitet så som emulator för android-enheter[19].

3.2 Amazon Web Services

Amazon Web Services, förkortat AWS, är en molntjänst med flera datacenter placerade över hela världen och flera miljoner kunder. AWS erbjuder flera olika tjänster som hjälper företag och privatpersoner att bland annat skapa molntjänster [20]. I detta avsnitt kommer de AWS-verktyg som använts i detta arbete att presenteras.

3.2.1 Lambda

Det verktyg som kommer att vara den stora beståndsdelen av trafikljussimuleringen för detta arbete är AWS Lambda. Lambda kan användas för att skriva så kallade ”Lambda functions” som är kodfunktioner som kan anropas genom exempelvis ett HTTP anrop. Istället för att behöva hantera servrar och konfigurera dessa kan kod skrivas direkt hos utvecklaren, distribueras över till Lambda tjänsten som ligger i molnet, och sedan köras när den anropas. Utvecklingen kan därför gå väldigt snabbt

och enkelt. Lambda-utvecklaren slipper sköta underhåll av servrar och det blir för vissa lösningar billigare eftersom Lambda endast kostar när funktionen körs [21].

Lambda funktioner kan skrivas i flera olika språk, bland annat Python 3 som valdes för detta projekt.

3.2.2 API-Gateway

API-Gateway är en tjänst inom AWS för att ansluta applikationer och tjänster utanför AWS till resurser så som servrar och funktioner skapade i molnet [22]. API-Gateway skapar RESTful (se 3.2.3) APIer som är HTTP-baserade och som kan implementera alla standardmetoder för HTTP (GET, POST, PUT, PATCH och DELETE). Inom API-Gateway så går det sedan att konfigurera tjänsten så att anrop till en specifik webbaddress tillsammans med tillhörande HTTP-metod startar en respons från en bestämd AWS-tjänst, till exempel en lambdafunktion.

3.2.3 RESTful API

RESTful API är ett gränssnitt som används för att säkert kunna utbyta information mellan två system över internet [23]. RESTful följer mjukvaruarkitektuern REST (Representational State Transfer) som följer följande principer:

- **Enhetligt gränssnitt** - anrop till servern samt svar från servern följer ett standardformat.
- **Tillståndslöst** - varje förfrågan hanteras helt isolerat från tidigare anrop vilket innebär att servern kan förstå och utföra varje förfrågan varje gång.
- **Användning av lagersystem** - autentiserade mellanhänder kan användas för att komunicera mellan klient och server.
- **Cachebarhet** - API-svaren anger om visst innehåll är cachebart eller inte för att minska mängden repeterat data som skickas från server till klienten genom att klienten i stället kan hämta innehåll från sin cache.
- **Stöd för kod mot begäran** - servern kan skicka ut kod till klienten som tillfälligt utökar eller anpassar innehållet för klienten genom att skicka kod som klienten kan köra.

Fördelar med att använda RESTful API är att det enkelt går att skala upp och ner applikationen och att servern inte behöver överbelastas med tidigare klienters anrop då det är tillståndslöst [23]. Det erbjuder även en flexibilitet då servern och klienten kan utvecklas parallellt med varandra.

4

Genomförande

Detta projekt har bestått utav flera separata delar som tillsammans skapat arbetsflödet för projektet. I detta kapitel presenteras dessa separata moment.

4.1 Insamling av användarkrav

VeloFlow ska i slutändan kunna användas av cyklister som är vana vid applikationer som är enkla att använda och interagera med även när de aktivt cyklar. För att kunna förstå användarens behov och sätta kriterier som behöver uppfyllas av VeloFlow så utfördes användartester tidigt i projektet.

4.1.1 Upplägg av användartester för användarkrav

Användartesterna utfördes med hjälp av intervjuer och observationer där testpersoner fick cykla korta sträckor medan de på olika vis interagerade med en smartklocka. Varje gång testpersonerna cyklade sträckan så fick de utföra olika uppdrag. Dessa uppdrag var bland annat att känna av vibrationer från klockan, lyssna på ljudklipp från hörlurar eller att interagera med klockan genom att läsa på skärmen eller att svepa med fingret över touch-skärmen. När testpersonen utfört uppdraget fick den svara på frågor angående hur svårt uppdraget var och hur distraherande det var för cyklingen. Vi som administrerade testerna dokumenterade även våra observationer av testpersonerna medan de cyklade. Aspekter så som hur mycket testpersonen vinglade med cykeln under tester och hur fokuset såg ut att påverkas skrevs ner för att sedan kunna jämföras med testpersonernas egna uppfattningar om hur testerna gick.

Testerna utfördes utav sex stycken olika testpersoner och det tog cirka 20 till 30 minuter per testperson att genomföra alla tester. Här är listan på de tester som genomfördes:

1. Känna av långa vibrationer från klockan

2. Känna av korta vibrationer från klockan

3. Läsa av tiden på klockan
4. Läsa av ett meddelande som skickas till klockan
5. Lyssna på ett ljudklipp där en röst kan höras genom Bluetooth hörlurar
6. Känn av ett vibrationsmönster bestående utav långa och korta vibrationer
7. Utför touch-gester på klockans skärm
8. Använd de fysiska knapperna på klockan
9. Utför rörelse-gester med klockan (testpersonen gavs exemplet att skaka på klockan)
10. Prata in i klockans mikrofon
11. En scenariofråga

Det sista testet, scenariofrågan, bestod utav att testpersonen cyklade rakt fram mot ett igenom scenariot påhittat trafikljus medan en av testadministratörerna beskrev ett scenario. Först var det att det påhittade trafikljuset var grönt och att testpersonen skulle fantisera att klockan började vibrera snabbare och snabbare. Det andra scenariot var igen att det påhittade trafikljuset är grönt men att testpersonen nu skulle fantisera att klockan började vibrera långsammare och långsammare. Syftet med scenariofrågan var att testa vad intuitionen sa till testpersonerna och om det fanns ett mönster i intuitionen att utnyttja.

4.1.2 Testmiljön

Testmiljön var en riktig trafiksituation, men relativt säker. För att undvika eventuella olyckor som skulle kunna ske om testpersonen blir för distraherad i riktig trafik så utfördes alla tester på en kortare raksträcka som var fri från biltrafik och hade minimal cykeltrafik. För majoriteten av testerna hyrdes en cykel genom Styr och ställ-tjänsten som finns i Göteborg [24]. Smartklockan som användes för testerna varierade något, men de gemensamma faktorerna var att klockorna var av märket Garmin, hade nästan samma storlek, satt på handleden, hade vibrationsfunktion samt en digital skärm. Applikationen som skapas för detta projekt har Wear OS som målplattform, inte Garmin, men för dessa tidiga användartesterna krävdes endast grundläggande smartklockefunktioner som både Wear OS klockor och Garmin-klockor har. Beslutet togs därför att Garmin-klockor skulle fungera likvärdig i utvärderingen av interaktioner med smartklockor och eftersom det fanns en större tillgång till Garmin-klockor inom projektgruppen så användes dessa. När ljud skulle spelas upp så hörde testpersonerna dessa genom ett par Bluetooth-hörlurar.

Målet var att göra testet så säkert som möjligt, men detta skapar även vissa begräns-

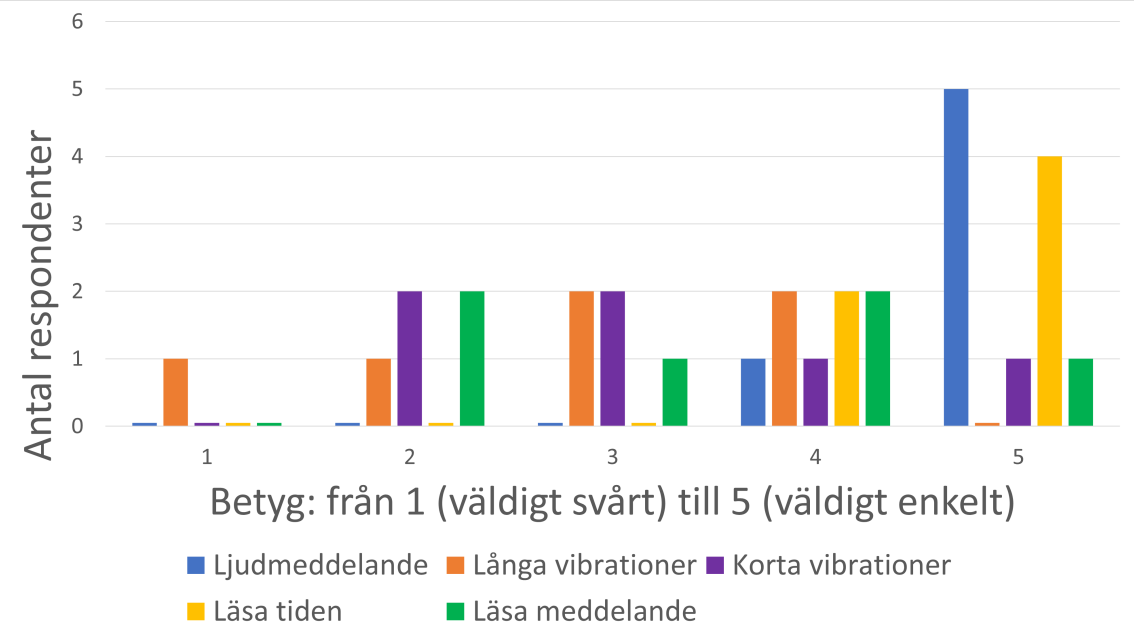
ningar som kan påverka datan. En riktigt trafikmiljö kan innehålla fler trafikanter att fokusera sin uppmärksamhet på och riktiga trafikljus och korsningar som behöver navigeras. Dessa komponenter hade gjort det svårare att interagera med klockan och det finns därför en risk att datan som samlas in under detta användartest inte hade sett likadan ut i en verklig situation.

4.1.3 Genomförande och resultat av användartester

Användartesterna bestod som tidigare förklarat av flera mindre tester, och flera av dessa gav användbar data som kunde användas vid designval för VeloFlow och implementationen av dess funktioner. Testerna kan delas upp i kategorier, där olika kategorier har olika påverkan på applikationens design. De två kategorierna som testerna delades upp i är *klocka till användare-interaktion* och *användare till klocka-interaktion*.

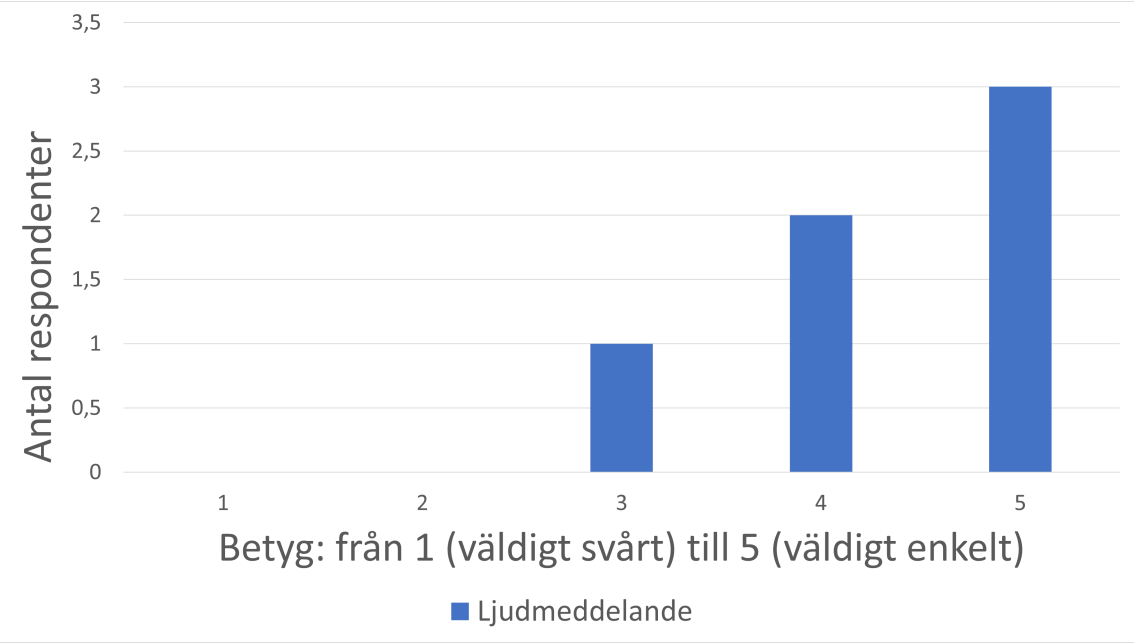
4.1.3.1 Klocka till användare-interaktion

Den första kategorin fick namnet *klocka till användare-interaktion* och gav information angående vad som fungerade bra och mindre bra när smartklockan skulle få bärarens uppmärksamhet. Testerna bestod av att klockan vibrerade med korta vibrationer, långa vibrationer och en kombination av korta och långa vibrationer. Sedan visade klockan text på klockan, visade tiden på klockan och spelade upp ljudmeddelanden genom hörlurar.



Figur 4.1: Resultat för användartester, klocka till användare-interaktion. Denna tabell visar hur enkelt det var för testpersonerna att uppfatta de olika interaktionerna.

Ljudmeddelande Inom kategorin *Klocka till användare-interaktion* var det ljudmeddelanden som gav bäst resultat. I figur 4.1 går det att se att testpersonerna

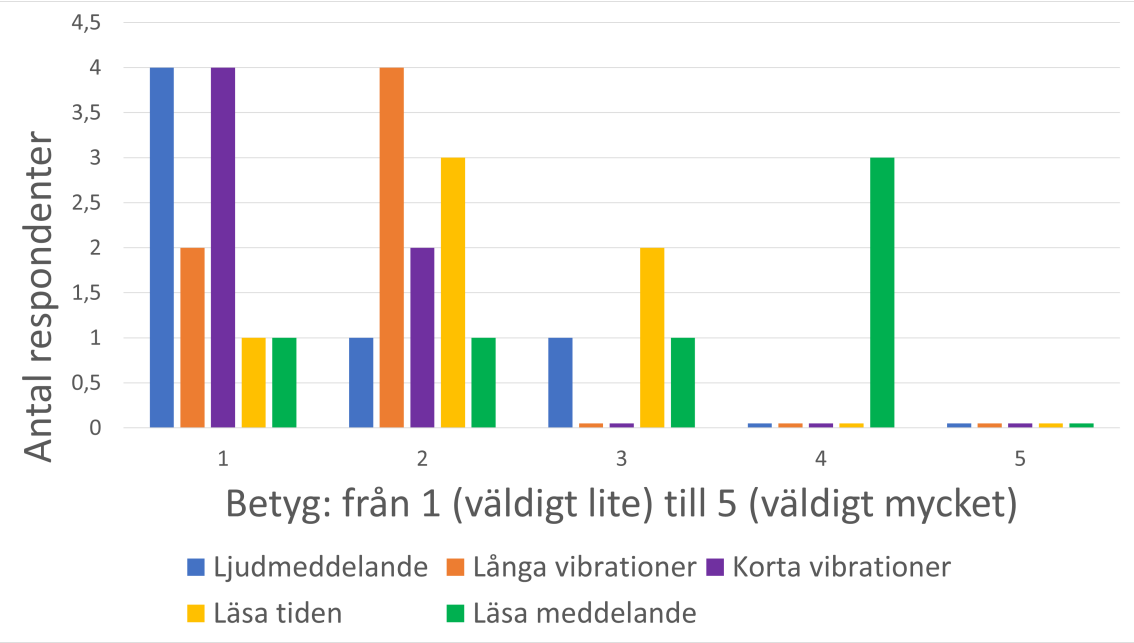


Figur 4.2: Resultat för användartester, klocka till användare-interaktion. Denna tabell visar hur enkelt testpersonerna tyckte det var att förstå informationen i ljudmeddelandet

kände att ljudmeddelandet var mycket enkelt att höra. Det gäller inte bara att höra meddelandet utan även att förstå vad som sas i det, och även detta gav testpersonerna höga poäng på vilket kan ses i figur 4.2. Dessutom var ljudmeddelandet inte så distraherande, utan testpersonerna kände att deras fokus fortfarande kunde hållas på deras omgivning. Det kan ses i figur 4.3. Våra observationer kunde även understryka detta resultat. Slutsatsen kunde därför dras att ljud på något sätt borde användas i VeloFlow.

Vibration Att ljudmeddelande fungerade bäst gick emot vissa förväntningar av att vibration var det som skulle fungera allra bäst och skulle ha förmågan att användas som det enda sinnesintrycket, men från användartesterna så kunde det tydligt synas att vibrationer var sämre på att få testpersonernas uppmärksamhet och de var svårare att tyda. Resultatet kan läsas av i figur 4.1. Vibrationerna var trots det inte så distraherande som kan ses i figur 4.3 och beslutet togs därför att behålla dem som ett stöd till ljudsignalerna. Att implementera så många olika sinnesintryck som möjligt kan ge användaren möjlighet att välja sin favorit och breddar användargruppen då alla användare kan få uppfylla sina unika behov.

Grafiska intryck Alla kommunkikationsvägar är trots det inte optimala även fast vissa användare kanske skulle föredra dem eftersom de kan vara opålitliga eller till och med farliga. Att använda skärmen på klockan för att förmedla information visade sig vara rätt så enkelt och tydligt för testpersonerna, men det var också väldigt distraherande. Detta resultat visas åter igen i figur 4.1 och 4.3. Ett inte så överraskande resultat, då det är viktigt att ha synen fokuserad på det runt omkring



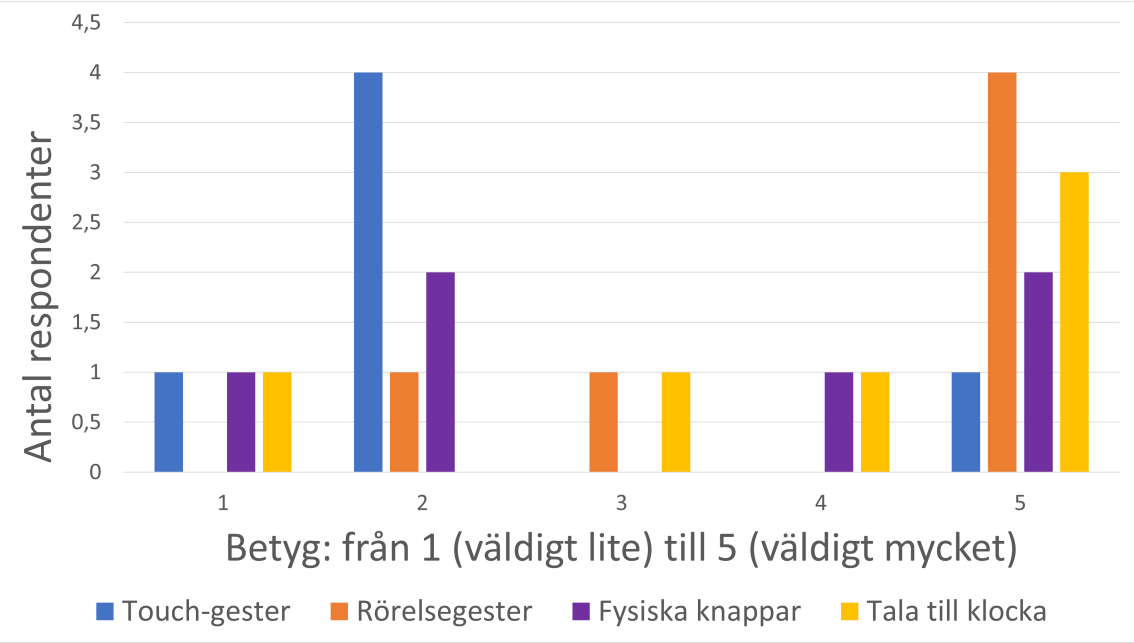
Figur 4.3: Resultat för användartester, klocka till användare-interaktion. Denna tabell visar hur distraherande testpersonerna tyckte att de olika interaktionerna var.

dig när du är i trafiken. Det finns trots det möjlighet att fortfarande använda visuella intryck utan att distrahera användaren allt för mycket. Det gick att se en skillnad i uppfattningsförmåga och distraktionsnivå mellan att läsa ett meddelande på klockan och att endast läsa av tiden på klockan. När testpersonerna läste av tiden på klockan uppfattades det inte som lika distraherande och det var enklare för dem att ta till sig informationen. Det är därför möjligt att ett grafisk gränssnitt skulle kunna användas i VeloFlow om det går att utforma på ett sätt som är väldigt tydligt och lättläst. En design som är intuitiv, som inte kräver för lång betänketid varje gång den läses av.

4.1.3.2 Användare till klocka-interaktion

Tester utfördes också för att se på vilka sätt användaren kan interagera med applikationen genom klockan. Det fanns inga klara planer på hur eller om användaren ska interagera med VeloFlow mer än för att få information, men om det krävs interaktion är det bra att ha data angående vilka sätt som fungerar bra eller mindre bra.

Testernas utformning var likadan som för *klocka till användare-interaktion* testerna, det vill säga att testpersonerna fick cykla en kort sträcka i en säker miljö medan de bar på en smartklocka. Under dessa tester fick testpersonen inte information eller intryck från klockan, utan fick istället själv vara den som gav klockan information. Testpersonerna fick fritt testa att använda klockans touch-skärm, fysiska knappar, göra gester med armen som klockan satt på och prata in i klockan. Det fanns ingen gräns i mängden interaktioner de fick göra med klockan och det fritt för testpersonen att tolka hur de skulle göra interaktionerna. De fick sedan berätta hur enkelt det

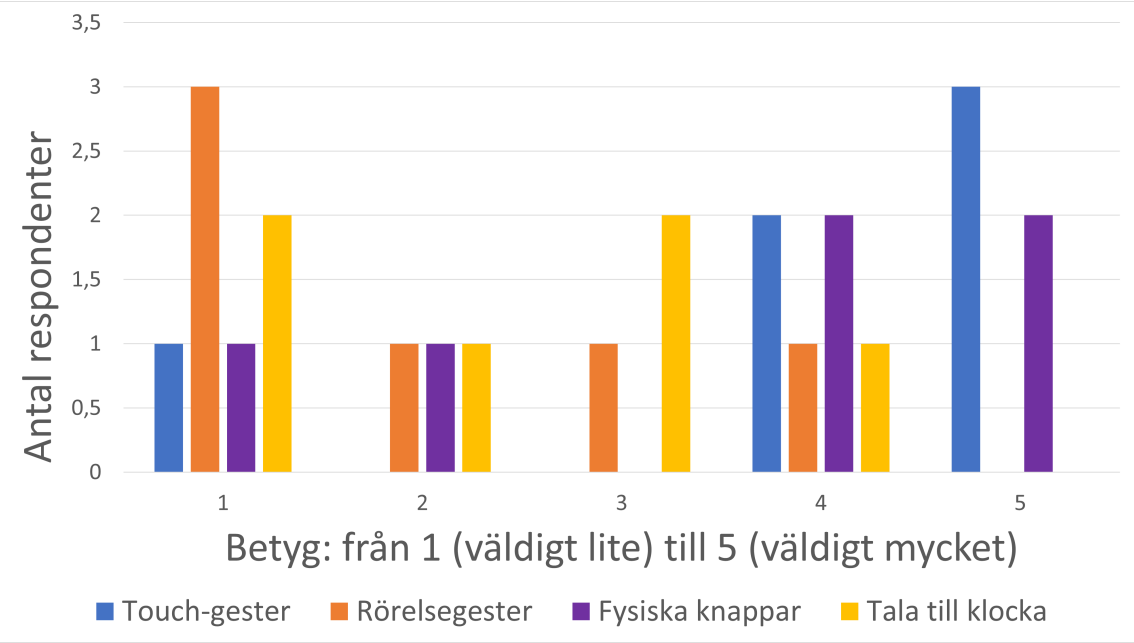


Figur 4.4: Resultat för användartester, användare till klocka-interaktion. Denna tabell visar hur enkelt det var för testpersonerna att utföra interaktionerna.

var att utföra dessa tester och gav betyg på hur distraherande dessa handlingar var för deras cykling.

Det test som gav sämst resultat var testet med touch-skärmen. Det var både svårt och väldigt distraherande för de flesta testpersonerna att göra touch-gester på klock-skärmen, se resultat i figur 4.4 och 4.5. Observationerna som antecknades stödjer även detta resultat, då det tydligt gick att se på de flesta testpersonerna att deras cykling blev mer ostadig. Några av testpersonerna gav även ytterligare kommentarer vid detta test, som var att det är jobbigt att släppa händerna från styret medan man cyklar då alla inte är vana vid att cykla utan. Om man inte släpper handen på sidan som klockan sitter utan istället endast släpper en av händerna så påpekade en testperson att det blir en svår vinkel att utföra touch-gester på klockan.

De andra testerna, att använda de fysiska knapparna på klockan, göra rörelsegester med klockan och att prata in i klockan gav bättre resultat men fortfarande inte på en nivå som skulle kunna bevisa att den interaktionen med klockan är ett bra sätt för alla cyklister. Det test som gav bäst resultat var att göra rörelsegester med klockan.



Figur 4.5: Resultat för användartester, användare till klocka-interaktion. Denna tabell visar hur distraherande testpersonerna tyckte att de olika interaktionerna var.

4.2 Utveckling av applikationen

Parallellt med användartesterna så påbörjades utvecklingen av VeloFlow. I detta avsnitt presenteras olika val som gjorts under utvecklingen av applikationen och anledningen till dessa val.

4.2.1 Foreground service

För att VeloFlow ska fungera korrekt så är det viktigt att tillförslen av platsdata aldrig avbryts medan användaren är ute och cyklar och att vibrationer och ljudmeddelanden alltid kan aktiveras oavsett om skärmen på klockan är släckt. Detta är överraskande svårt att garantera om det inte finns tidigare erfarenhet av att utveckla Android-applikationer för smartklockor. Vid testning av applikationen upptäcktes det tidigt att skärmen på smartklockan snabbt släcks om användaren inte aktivt rör vid den, och då slutar även applikationen att hämta platsinfo. För att få platsinformation används ett API som finns tillgängligt för Android-applikationer som heter Fused Location Provider [25]. Det är detta API som slutade ge uppdateringar om klockan gick in i sitt strömsparläge kallat ”Ambient Mode”. Det går att hålla element synliga på klockan även i ambient mode, men detta kan kraftigt påverka batteritiden för klockan så det avråds [26]. Det var även osäkert om detta skulle hålla Fused Location Provider från att bli inaktiv och då sluta ge uppdateringar om användarens position och om det fortfarande skulle gå att göra vibrationer och spela upp ljud. Lösningen blev därför att använda sig av ett Android-verktyg kallat ”Foreground service”. Denna tjänst sätts igång av applikationen och fortsätter att köras

tills dess att applikationen stänger av den [27]. På det sättet kan platsinfo alltid hämtas och vibration och ljudmeddelanden kan alltid spelas, även om applikationen inte är framme på klockans skärm. Att ha en foreground service som ständigt ligger och arbetar i bakgrunden kommer trolig att påverka batteritiden negativt för klockan, men eftersom det är en service som är nödvändig för applikationens funktionalitet så måste det användas.

4.2.2 Design av ljudmeddelanden och vibrationer

Då användarstudierna visade att de effektivaste sätten att förmedla information till cyklister, utan att vara för distraherande, var att använda ljudsignaler och vibrationer så skapades två stödtjänster för VeloFlow. Den ena tjänsten var för att skapa och spela upp ljudklipp på klockan och den andra för att skapa olika vibrationsmönster och vibrera dessa mönstren.

4.2.2.1 Testa ljud

Användaresterna visade att cyklister inte har några större problem med att få till sig information via tal genom att höra ett ljudmeddelande. Att använda sig av en röst som förmedlar information kunde därför ses som ett rimligt sätt att kommunicera med användaren med hjälp av ljud. Det går även att använda ljud på andra sätt, i form av enkla ljudeffekter som inte innehåller tal. Detta hade gjort applikationen mer tillgänglig till en större mängd användare då det inte skulle finnas en risk för språkbarriärer. Ljudeffekter hade även haft fördelen att de kan vara kortare och då ta upp mindre tid av cyklistens uppmärksamhet.

För att avgöra om ljudeffekter eller röstmeddelanden borde användas i denna första iteration av applikationen så utfördes några enkla tester där en grupp av 4 personer fick lyssna på åtta stycken ljudeffekter och fyra stycken ljudmeddelanden som sägs av både en kvinnlig och en manlig röst. Ljudeffekterna bestod av korta effekter med olika toner och mönster som skapades med hjälp av verktyget JSFXR[28]. Rösterna var engelska fraser som sas av syntetiska röster som skapades med hjälp av en ”Text to Speech” tjänst[29]. Testpersonerna fick höra alla ljudklipp och ge kommentarer om hur de upplevde klippen och vad de fick för intryck.

Testerna visade att ljudeffekterna inte tillförlitligt kunde förmedla samma intuitiva känsla till olika personer och de var generellt mer stressande. Ljudeffekterna var även svåra att skilja åt från varandra. Rösterna var däremot enkla att skilja åt mellan varandra och upplevdes tydligare. Med röstmeddelanden krävs inte heller en instruktionsguide för att börja använda applikationen eftersom eftersom rösterna förklarar sig själva.

I valet mellan kvinnlig och manlig röst föredrog en majoritet en manlig röst. Detta kan ha berott på kvalitén av de röster som fanns att använda i ”Text to Speech”-tjänsten.

4.2.2.2 Skapa ljudmeddelanden

Valet gjordes att använda röstmeddelanden, så fraser som kunde tänkas behövas togs fram. För att förbättra tydligheten av de röstmeddelanden som ska användas så skapades nya meddelanden som istället för en syntetisk röst använde en riktig röst. De spelades in i både engelska och svenska. Valet av språk är upp till användaren.

Stödtjänsten för ljudmeddelanden skapar en mediaspelare som kan spela upp ljudfiler. Så för att expandera applikationen med mera ljudmeddelanden så räcker det att lägga till en ljudfil i resurserna och skapa en metod i tjänsten som använder mediaspelaren för att spela upp ljudfilen.

4.2.2.3 Skapa vibrationer

För vibrationer så skapar stödtjänsten ett objekt av typen Vibrator, som sedan kan användas för generera vibrationer i klockan. För att skapa ett vibrationsmönster i denna stödtjänst så behöver två arrayer med samma längd skapas. Den första arrayen ska innehålla tider angivet i millisekunder och den andra arrayen ska innehålla amplituder. Tiderna i den första arrayen beskriver hur länge vibratorn ska vibrera med styrkan angivet av amplituden på motsvarande index i den andra arrayen. För amplituderna så är värdena 0-255 tillåtna, där 0 motsvarar att vibrationsmotorn i klockan är avstängd och 255 att motorn vibrerar med full styrka, motorn kommer alltså att vibrera med styrka enligt ekvation 4.1. Utöver de två arrayerna så behöver ytterligare en siffra anges för att bestämma om vibrationsmönstret ska upprepas eller enbart köras en gång. För de vibrationer som skapades så gjordes valet att alltid använda amplituden 255, då det upplevdes svårt att känna av vibrationerna när de var för svaga.

$$Vibrationsstyrka = \frac{amplitud}{255} \cdot Maximal\ vibrationsstyrka \tag{4.1}$$

4.2.2.4 Användartester för vibration

För att undersöka vilka typer av vibrationer som är enkla att känna igen och vilka associationer som olika vibrationer ger upphov till så genomfördes en halvstrukturerad intervju på fyra användare. Som introduktion till användartesterna fick användarna tänka sig att de var ute och cyklade och de blev ombedda att svara med detta som premiss. Under intervjun fick användarna känna olika vibrationsmönster och sedan förklara för varje vibrationsmönster vad de upplever att vibrationen vill förmedla för information. Efter att alla vibrationsmönster hade testats så fick användarna fritt kommentera de vibrationer de känt, samt ge förslag på ytterligare mönster de tror hade varit effektiva att använda. I figur 4.6 visas de vibrationerna som testades av användarna.

Utifrån intervjuerna efter användartesterna framkom att det för vibrationsmönster 1, figur 4.6a, var otydligt vad den skulle betyda. En av användarna trodde att vibrationsmönstret indikerade att sänka hastigheten samtidigt som en annan användare tyckte att mönstret var stressande och att den indikerade att hastigheten

skulle höjas. Den upplevdes också svår för vissa att känna av då den hade korta vibrationer.

För vibrationsmönster 2, figur 4.6b, så upplevde alla användare vibrationsmönstret som långt och vissa kände att detta mönstret upplevdes som om det var någon som ringde. Vibrationsmönstret var enligt användarna enkelt att känna av tack vare de längre vibrationerna.

Vibrationsmönster 3, figur 4.6c, upplevdes av majoriteten som om ett inkommande samtal vilket upplevdes irriterande när det inte fanns något samtal.

Det fjärde vibrationsmönstret, nr 4 som kan ses i figur 4.6d, upplevdes som otydligt och svårt att hänga med i exakt hur mönstret gick. Detta vibrationsmönstret upplevdes till skillnad från tidigare mönster som mer unikt och ingen associerade mönstret med ett inkommande samtal.

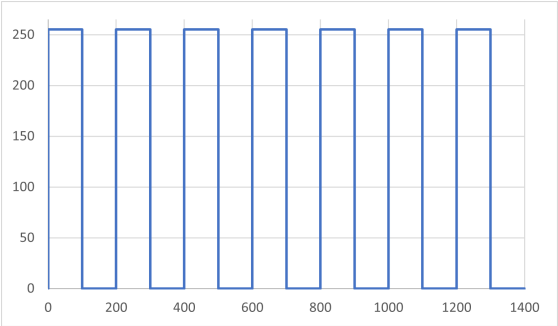
Det två sista vibrationsmönstren 5 och 6, figur 4.6e och figur 4.6f, upplevdes efter att användarna hade känt av båda som att vibrationsmönster 5 var menat att indikera att hastigheten skulle sänkas och att vibrationsmönster 6 var tänkt att informera användare om att de skulle accelerera eller skynda på.

Vibration 5 och 6 upplevdes enkla att separera och gav intuitivt en känsla av att sin hastighet borde förändras, antingen snabbare eller långsammare. Dessa vibrationer blev därför utvalda till att implementeras in i applikationen.

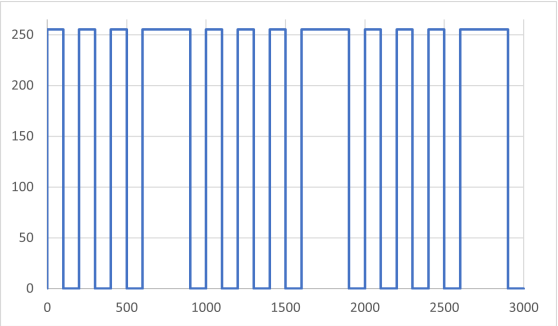
Ytterligare kommentarer som inkom var att det är viktigt att de vibrationer som används särskiljer sig från de som vanligtvis används för notiser för meddelanden och samtal.

4.2.2.5 Designval för ljud och vibrationer - summering

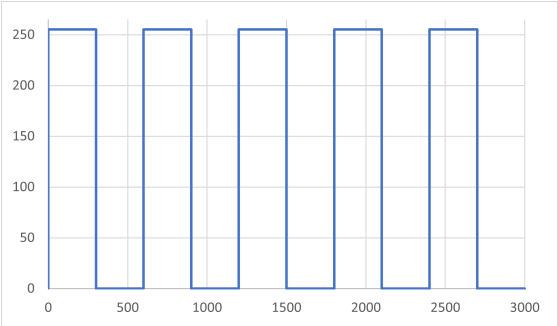
Med hjälp av användarstudier gick det att komma till slutsatser i vilka designval som borde göras för VeloFlows ljud och vibrationer. Intalade ljudmeddelanden fungerade bättre än ljudeffekter då de kan innehålla enkel information som inte behöver förlita sig på inlärning och viss intuitionsförmåga, och de blev ännu tydligare när en riktig röst användes i inspelningarna istället för en syntetisk maskinröst. Vibrationerna utvärderades och det framgick att nr 5 och 6 fungerade bäst då de gick att känna skillnad på dem och de gav en intuitiv känsla som kan utnyttjas i applikationen. Med dessa resurser kunde fortsatt arbete på VeloFlow utföras och det fanns en förhoppning om att applikationen skulle kunna ge tydliga instruktioner till användaren.



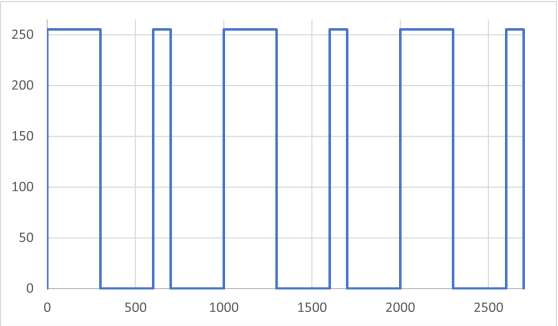
(a) Vibrationsmönster 1



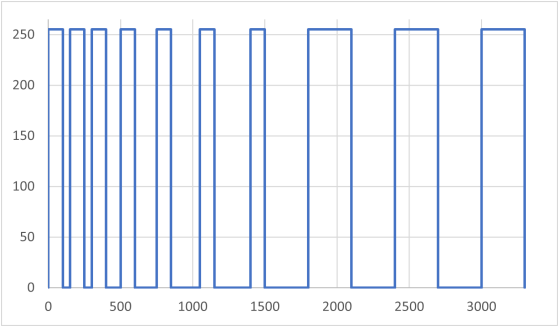
(b) Vibrationsmönster 2



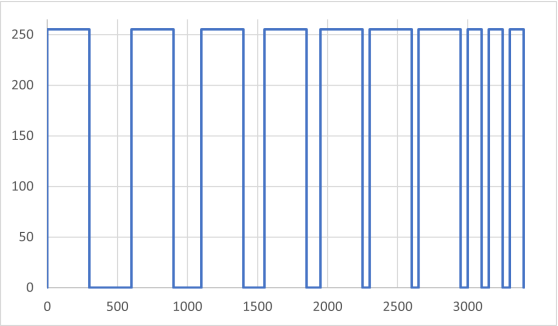
(c) Vibrationsmönster 3



(d) Vibrationsmönster 4



(e) Vibrationsmönster 5



(f) Vibrationsmönster 6

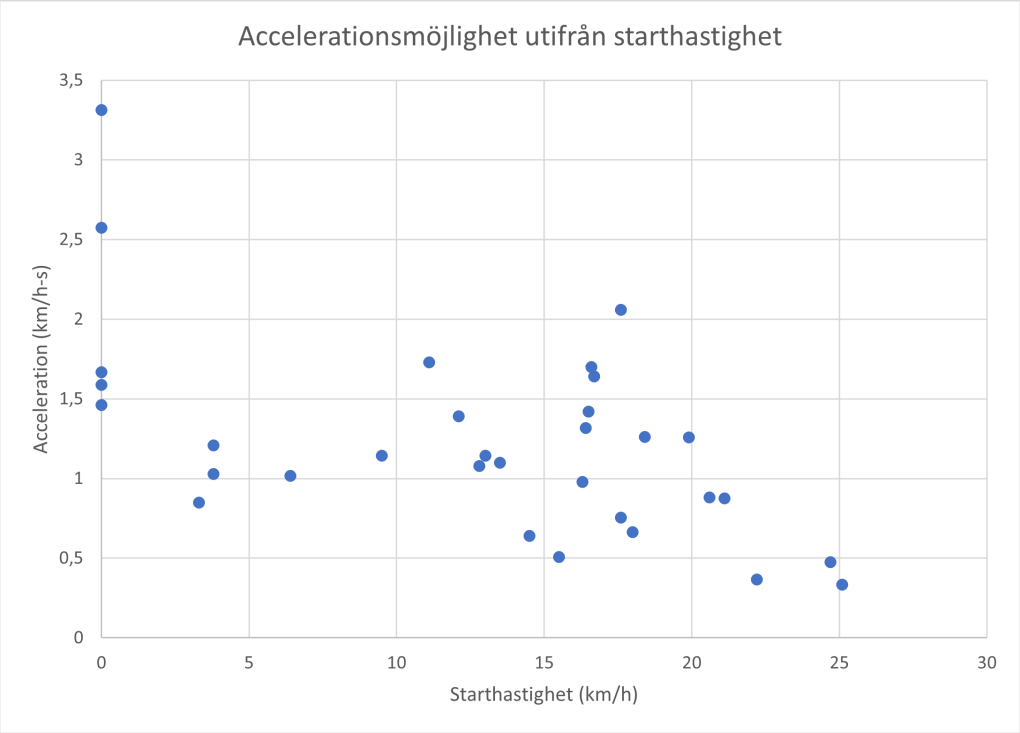
Figur 4.6: Vibrationsmönster som användes under användartester där y-axel visar styrkan på vibrationen och x-axeln visar tid angivet i millisekunder.

4.2.3 Utveckling av handlingsmodellen

För att komma fram till en logisk modell som ska kunna ge korrekta instruktioner till användaren så utvecklades applikationen iterativt där varje iteration testades av utvecklarna för att utvärdera den. Efter varje utvärdering gjordes utvecklingar på modellen för att göra den mer avancerad och exakt.

Ett verktyg som användes för modellutvecklingen var testning utav acceleration. För att få fram korrekt accelerationsförmåga hos genomsnittliga cyklister så utfördes accelerationstester utav utvecklarna då det inte gick att hitta trovärdig data om detta från andra källor. Det krävdes nämligen inte bara en generell accelerationsförmåga eller faktor, eftersom en cyklists accelerationsförmåga ser olika ut beroende på aktuell hastighet. Att accelerera kraftigt från nästan stillastående är enklare än att

göra en acceleration i närheten av sin maxhastighet. Testerna, vars resultat kan ses i figur 4.7, gav användbar data som möjliggjorde fortsatt utveckling utav applikationsmodellen.



Figur 4.7: Accelerationsmöjlighet utifrån starthastighet

Andra inkrement utav utvecklingen gav varierande distanser som modellen använder för att avgöra om användaren är tillräckligt nära för att få instruktioner, där distansen varierar beroende på användarens aktuella hastighet.

4.3 Molntjänsten

För att testa och utvärdera applikationen i detta arbete krävs det trafikljus att navigera och hantera. Tidigt i arbetet påbörjades en konversation med Stadsmiljöförvaltningen i Göteborg för att se över möjligheten att få tillgång till riktigt trafikljusdata. Ett arbete för att skapa en sådan lösning har påbörjats för autonoma fordon, men det var inte möjligt att ordna en lösning för vår VeloFlow inom den tidsram som fanns för detta arbete. Istället så behövdes en trafikljussimulering skapas.

För att i så stor utsträckning som möjligt efterlikna trafikljuslösningen efter hur en riktig källa för trafikljusdata skulle kunna se ut så bestämdes det att trafikljusen skulle simuleras genom en molntjänst, och alltså inte köras lokalt på klockan som kör applikationen. Data som behöver användas kan då hämtas via anrop genom internet, och simuleringen kan utvecklas parallellt med applikationen. Eftersom ingen tillgång

till riktig trafikljusdata funnits så finns det ingen garanti på att molnet som skapas för detta arbete perfekt kommer att likna Göteborgs verkliga system. Utifrån den information som Stadsmiljöförvaltningen gav så är det troligt att det åtminstone finns likheter och att skillnaderna inte är så pass stora att relativt lite omarbete skulle lösa skillnaderna.

I detta avsnitt förklaras hur molntjänsten i detta arbete utvecklades och varför vissa designval gjordes.

4.3.1 Trafikljussimuleringens komplexitet

För detta arbete krävs inte en simulering som är av samma komplexitet och storlek som ett trafikljusnät för en riktigt stad. Istället krävs endast något som ger en tillräckligt bra bild för att se om konceptet för VeloFlow kan hjälpa cyklister med att navigera trafikljus. Nivån av den krävda komplexiteten bestämdes vara att simuleringen behöver bestå av mer än ett ljus men inte fler än vad som går att cykla förbi under en testrunda, och att ljusen kan ha rött eller grönt ljus som status. Gult ljus är användbart för trafikanter då det förbereder denna på att antingen sakta in eller börja accelerera, men för applikationen krävs inte denna varning eftersom den utgår från tiden det är beräknat vara kvar för ljusets status.

Vid verkliga korsningar kan flera trafikljus vara placerade och dessa trafikljus påverkar varandra. Alla kan till exempelvis inte vara gröna samtidigt. För simuleringen för detta arbete krävs inte denna extra komplexitet för att utvärdera grundkonceptet angående om applikationen har potential att förenkla att få en grön våg. Därför kommer det inte att simuleras korsningar där flera trafikljus är placerade, utan endast ett trafikljus kommer att finnas vid varje punkt.

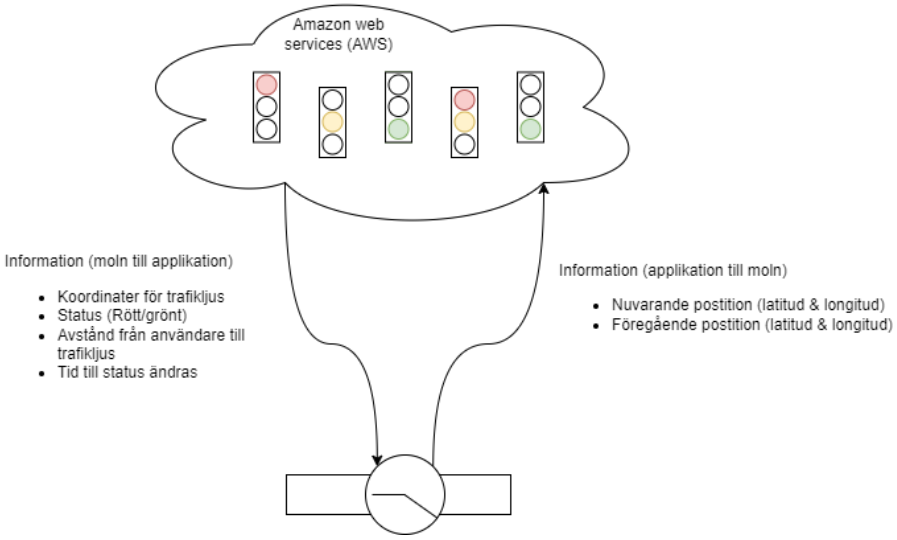
Ljusen behöver också ha riktiga koordinater för att det ska gå att testa applikationen i ett verkligt scenario. Genom att lägga ljusen längs en verklig väg kan applikationen testas på ett mer verklighetstroget vis.

4.3.2 API-flöde

Arbetsuppgiften och kraven som fanns för molntjänsten utgjorde grunden för vilken data som behöver kunna hämtas via API:t. Applikationen ska få information angående distans till nästkommande trafikljus, vilken status trafikljuset har och hur lång tid det är kvar tills det ska slå över. Anledningen till valet att molntjänsten ska skicka avstånd och inte position för trafikljuset är för att minska arbetslasten för klockan som applikationen körs på genom att istället utföra avståndsberäkningarna i molnet. Molntjänsten som AWS erbjuder drivs av kraftfullare hårdvara än klockan som applikationen körs på. Extra information som kan ge värde vid felsökning är att få koordinaterna för trafikljuset vars information skickas för att säkerställa att korrekt trafikljus står i fokus.

För att API:t ska kunna skicka ut denna information så behöver VeloFlow skicka an-

vändarens nuvarande position och föregående position i form av koordinater. De två positionerna behöver skickas för att avgöra användarens riktning, denna beräkning förklaras vidare i kommande del 4.3.3.



Figur 4.8: Datautbyte mellan moln och applikation. Pilarna till molnet representerar den data som skickas till molntjänsten från applikationen på klockan. Pilarna till klockan visar svaret som molnet genererar till applikationen.

Det totala datautbytet mellan applikationen på klockan och Lambda funktionen i AWS visualiseras i bild 4.8. Där visar pilarna från klockan till molnet den information som skickas till Lambda funktionen. Pilarna som går från molnet till klockan är svaret som applikationen får tillbaka och som sedan utgör grunden för det val som ska göras för användaren.

4.3.3 Beräkningar

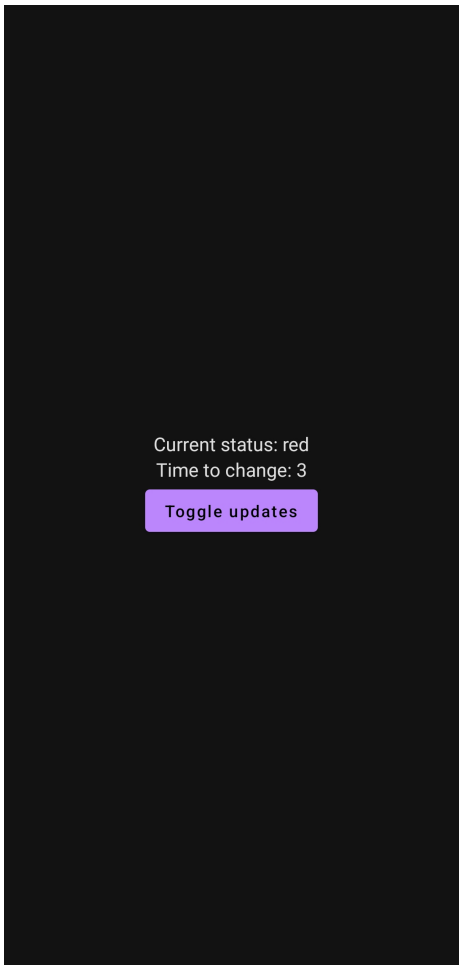
Det viktigaste problemet som molnet löser är att hitta det närmsta trafikljuset som användaren är på väg mot. För att få fram detta sorterar molnet listan av trafikljus efter hur långt ifrån det är användarens position, och tar sedan första ljuset från listan. Datan angående användaren nuvarande position och föregående position används sedan för att se om användaren är på väg mot detta trafikljus. Om distansen minskat från ljuset med nuvarande position jämfört med föregående position så vet molnet att användaren är på väg mot det valda ljuset och kan då skicka information gällande detta ljus. Om så inte är fallet så tas nästa ljus i den sorterade listan och samma kontroll utförs igen tills ett ljus hittats. Om användaren inte är på väg mot något av ljusen i listan så skickas statusen ”none” för att signalera att inget trafikljus står i sikte.

Distansberäkningen sker genom en matematisk uträkning som använder koordinater för att få ut avståndet i meter. I uträkningen finns det en felmarginal på cirka en meter.

4.4 Mätning av batteriförbrukning

medan applikationens funktionalitet testades så antecknades även batteriförbrukningen. Applikationen kräver konstant GPS-data medan den kör, och det hade då varit intressant att se hur mycket detta påverkar batterilivslängden hos en laddning. Batteriförbrukning under normal användning av klockan och batteriförbrukningen under ingen användning av klockan antecknades också för att ha data att jämföra med. Resultatet från dessa mätningar presenteras i avsnitt 5.3

4.5 Stödapplikation



Figur 4.9: Gränssnitt för stödapplikationen

Personen som har på sig smartklockan kan nu få information om trafikljus och vad den bör göra för att hinna med grönt ljus, men för att tillförlitligt kunna utvärdera applikationen och se om personen som använder applikationen lyckas att passera ett virtuellt trafikljus så hade det varit bra om en åskådare kan få information angående trafikljusens status. För att lösa detta skapades en simpel stödapplikation som kommer göra testningen och utvärderingen av huvudapplikationen mer exakt. Stödapplikationen, vars gränssnitt kan ses i figur 4.9, förmedlar aktuell status på de simulerade trafikljusen och hur lång tid det är kvar tills de slår över. Statusen hjälper den som observerar att se om det var grönt när personen på cykeln passerade platsen för ett simulerat trafikljus, och tiden kvar för omslag hälper även observatören att se om lämplig handling föreslås av applikationen.

Applikationen skapades på samma sätt som huvudapplikationen för detta arbete, det vill säga i Android studio skrivet i Kotlin med Jetpack Compose som ramverk. Skillnaden är endast att stödapplikationen skapades för Android-telefoner och inte Wear OS klockor.

Med stödapplikationen till hands kunde nu slutanvändartester utföras så att data på ett pålitligt sätt kan samlas in, då testpersonerna inte själva behöver titta på klockan för att avgöra om de hann i tid till ett simulerat trafikljus eller inte. Den uppgiften kan nu istället delegeras till testledarna.

Figur 4.10: Exempel på hur ett slutanvändartest kunde se ut. Personerna på bilden är studenterna som gjorde detta arbete

4.6 Slutanvändartester

När VeloFlow utvecklats och justerats till den punkt att den kunde ge vad som verkade som korrekta kommandon så avslutades utvecklingen. Det finns mer arbete som skulle kunna utföras på applikationen och fler funktioner som skulle kunna implementeras men vid detta stadie så ansågs VeloFlow vara så pass funktionell att det går att utvärdera om idén är rimlig och om designen är gjord korrekt. Det var därför dags att påbörja slutanvändartesterna, som ska hjälpa till med att svara på detta arbetets frågeställningar.

4.6.1 Upplägg för slutanvändartester

Denna sista användarstudie lades upp som en strukturerad intervju tillsammans med observationer, så som den första användarstudien som användes för att samla in användarkrav. Den strukturerade intervjun bestod av frågor som utvärderade vad testpersonerna tyckte om tydlighet, distraktion, vilja att fortsätta använda VeloFlow och generell upplevelse. Observationerna som togs hjälpte till att styrka datan om hur applikationen påverkade testpersonerna, och användes även för att se hur många gånger testpersonerna lyckades passera ett simulerat trafikljus medan det var grönt.

4.6.2 Genomförande av slutanvändartester

Slutanvändartestet utfördes med VeloFlow installerad på en WearOS smartklocka och fyra simulerade trafikljus placerade längs ett cykelstråk. Exakta instruktioner för användartester kan ses i bilaga A. Testet utfördes utav 5 personer och tog cirka 20 minuter att utföra. Personerna fick använda en Styr och Ställ-cykel som hyrdes inför varje test. De fick bära hjälm för säkerhetens skull, men testet utfördes vid ett cykelstråk som endast korsade en trafikerad bilväg och därför ansågs vara säker.

I resultatdelen av denna rapport, i avsnitt 5.2, presenteras resultatet av denna användarstudie.

5

Resultat

Detta kapitel lyfter resultaten från projektets genomförande. Fokus ligger på att presentera den färdiga applikationen, presentera resultaten från de avslutande användarstudierna och att ge svar på vår forskningsfråga: Vilka designfaktorer är viktiga när man skapar en smartklockeapplikation som underlättar för cyklister att få en grön våg?

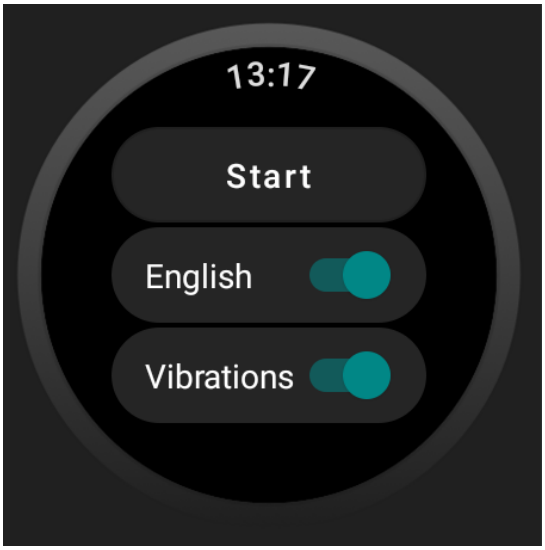
5.1 Applikationsbeskrivning och demonstration

Den färdiga applikationen, VeloFlow, har ett antal olika vyer eller skärmar med olika information. Den första skärmen, hemskärmen, kan ses i figur 5.1 och är vad användaren ser först när hen startar upp applikationen. På hemskärmen finns det två stycken av/på-knappar som används för att välja inställningar. Den första av/på-knappen används för att välja språk mellan svenska och engelska, den andra av/på-knappen är till för att aktivera/avaktivera vibrationer. Utöver de två inställningsknapparna så finns det en knapp för att starta igång huvudprogrammet, när denna klickas så byter applikationen skärm till skärmen i figur 5.2 som kommer att vara aktiv fram tills dess att applikationen har lyckats koppla upp sig molnet och användarens position har identifierats.

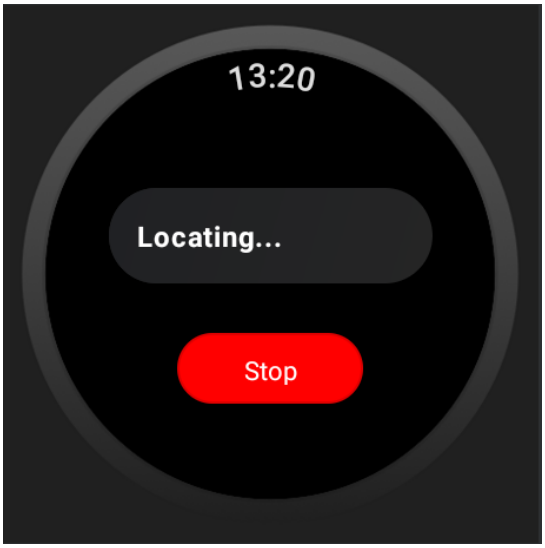
När VeloFlow lyckats lokalisera användaren så kommer den primära aktivitetsskärmen figur 5.3 att visas. På denna skärm visas information om användarens cykling. Den information som visas är cyklistens nuvarande hastighet samt status och avstånd till det i användarens riktning närmsta trafikljuset om användaren cyklar mot ett trafikljus. Skulle användaren inte vara på väg mot ett trafikljus så står status och avstånd som ”none”. Utöver information så finns det även en stoppknapp som tar användaren tillbaka till hemskärmen när den klickas.

Även om det på aktivitetsskärmen visas information så är VeloFlow designad för att användaren inte ska behöva titta på skärmen efter att den startat upp en aktivitet. När en aktivitet startas genom att klicka start på hemskärmen så startas en foreground service (se avsnitt 4.2.1) som börjar hämta platsinformation för användaren och kommunicera via API:t för vår Lambda-funktion som ligger i molnet. Kommunikationen mellan molnet och VeloFlow är uppbyggd så att applikationen skickar

användarens nuvarande position och användarens förra position, molnet kommer sedan ge ett av två möjliga svar. Det första svaret ges om användaren närmar sig minst ett trafikljus. I detta fallet så kommer molnet att skicka information om avståndet till det närmsta trafikljuset som användaren rör sig mot, vilken status detta trafikljuset har (rött eller grönt) samt hur långt tid det är kvar innan trafikljuset byter status. Det andra möjliga svaret från molnet skickas enbart om användaren är på väg bort från alla trafikljus i molnet, i detta fallet kommer molnet indikera att användaren inte är på väg mot något trafikljus genom att skicka svaret ”none”.



Figur 5.1: Hemskärm för applikationen



Figur 5.2: Aktivitetsskärm i väntan på lokalisering

Figur 5.3: Aktivitetsskärm under aktiv aktivitet

Givet att användaren är på väg mot ett trafikljus så kommer VeloFlow hjälpa användaren att anpassa sin hastighet så att användaren ankommer till trafikljuset när det är grönt. Applikationen gör detta genom att först se hur långt avstånd det är till trafikljuset och utifrån detta använda användarens hastighet för att beräkna hur lång tid det är kvar innan användaren ankommer till trafikljuset. Om den beräknade ankomsttiden är under en period när trafikljuset är grönt så kommer VeloFlow inte ge några rekommendationer. Om användaren däremot har en beräknad ankomsttid som är när trafikljuset är rött så kommer applikationen att ge rekommendationer till användaren. Vilken rekommendationen som användaren får beror primärt på trafikljusets nuvarande status, vilken hastighet som användaren håller, användarens maximala hastighet, nuvarande avstånd till trafikljuset och användarens möjlighet att accelerera.

Om det är grönt ljus och användaren har möjlighet att accelerera upp i en hastighet som gör att den kommer att hinna innan det slår över till rött så kommer VeloFlow att rekommendera detta. Om det däremot är så att användaren inte beräknas hinna komma upp i den hastighet som krävs för att hinna medan det är grönt så kommer applikationen i stället rekommendera att sänka hastigheten till en hastighet som gör att användaren kommer att komma fram lagom tills trafikljuset slår om till grönt igen.

Skulle det vara rött ljus så kommer VeloFlow i första hand undersöka om användaren hinner fram till trafikljuset innan det slår om till grönt, om detta är fallet så kommer rekommendation om att sänka hastigheten ges. Ifall användaren däremot är beräknad att komma efter det slår om till grönt säkerställer applikationen i stället att användaren inte är så långsam att det hinner slå om till rött igen. Skulle detta vara fallet så kommer applikationen se ifall det finns möjlighet för användaren att öka sin hastighet för att hinna inom tidspannet för nästa gröna ljus och rekommendera användaren att accelerera om det är möjligt.

Metoden VeloFlow har för att meddela användaren om vilken handling som användaren ska utföra är genom att spela upp en ljudsignal med uppmaningen ”Öka hastigheten” eller ”Sänk hastigheten”(eller motsvarande på engelska). Samtidigt så kommer klockan börja vibrera med en för varje uppmaning unikt vibrationsmönster. Vibrationerna kommer att upprepas tills dess att användaren uppnår den av applikationen planerade hastigheten eller tills dess att applikationen byter uppmaning. Utöver att vibrationerna upphör när användaren cyklar i rätt hastighet så kommer även en ljudsignal med meddelandet ”Hastighet uppnådd” att spelas upp.

5.2 Resultat slutanvändartest

Slutanvändartestet hade som uppgift att evaluera det byggda systemet och dess koncept, och det gick till på det sättet att ett antal testpersoner fick testa VeloFlow medan de cyklade längs en cykelsträcka. Denna cykelsträcka hade fyra simulerade trafikljus längst sig, och dessa trafikljus skulle personerna försöka åka förbi medan ljusen var gröna. Efter avslutad cykling fick testpersonerna några frågor att besvara. I detta avsnitt kommer resultaten från dessa sluttester att presenteras.

5.2.1 Testpersonernas förutsättningar

Slutanvändartestet utfördes utav 5 personer med varierande cykelvana. Den med minst cykelvana rapporterade att den aldrig cyklar men att den vet hur man gör, och den med mest cykelvana cyklar till sin arbetsplats flera dagar i veckan beroende på väder. Två var kvinnor och resterande tre var män, och flera nationella ursprung var representerade. Ingen av personerna uppgav någon sorts funktionvariation, så som nedsatt syn eller nedsatt hörsel. Ålder var spridd mellan 31 och 52 år.

5.2.2 Grön våg

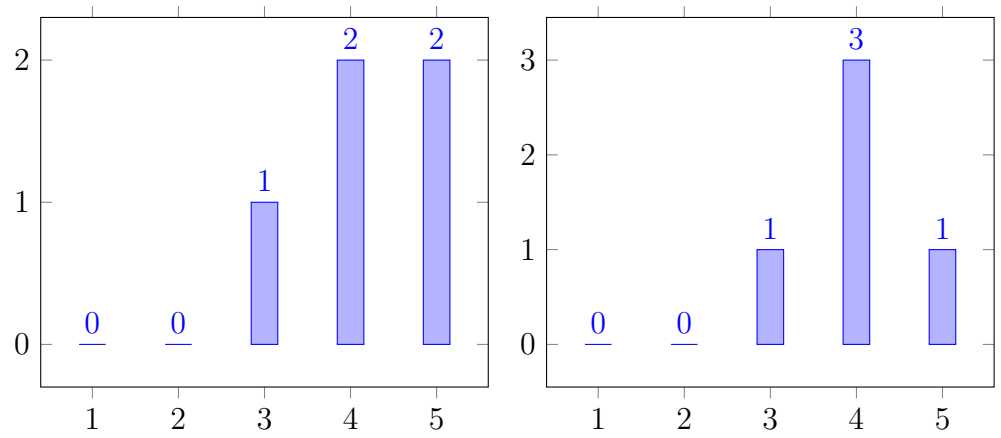
I sluttestet ingick det att räkna antalet lyckade passeringar förbi trafikljuset som fick namnet ”stubben” (Se bilaga A för förklaring av trafikljusnamnen). En lyckad passering innebär att användaren cyklade förbi trafikljuset medan det var grönt. Resultatet från dessa passeringar kan ge en indikation på hur väl VeloFlow kunde styra användaren till att korrekt passera trafikljus när de är i grönt läge utan att användaren själv har någon information om aktuell status. Under testet kunde inte testpersonerna själva se status på de simulerade trafikljusen utan kunde endast agera utifrån vad applikationen informerade dem om. Enligt en av punkterna för den första designtaktiken som presenterades i arbetet ”Co-riding With My eBike to Get Green Lights” [9] så testades här hur väl färdighetintegreringen mellan användaren och systemet fungerade. Testpersonen fick ingen information från omvärlden utan måste förlita sig på applikationen. Testpersonerna kunde under testets gång reglera hur mycket de ville bidra till processen att följa applikationen instruktioner för att hinna med de gröna ljusen, och VeloFlow reglerade då de instruktioner som den gav testpersonerna. På detta sätt kunde testpersonerna få bättre förståelse av applikationen och systemet, vilket ledde till bättre resultat i passeringarna. Detta är inte något som testpersonerna själva kommenterade på, men resultaten i passeringarna blev för varje testperson bättre ju längre tid de hade använt VeloFlow.

I tabell 5.1 står antalet lyckade passeringar för varje testperson.

Tabell 5.1: Antal passeringar gjorda av testpersonerna och hur många som var lyckade

Testperson	a	b	c	d	e
Totalt antal passeringar	4	4	4	4	5
Antal lyckade passeringar	3	3	3	4	5

Totalt utav alla passeringar som utfördes av testpersonerna så var 85.7% av dessa passeringar lyckade. Det bör påminnas att VeloFlows syfte inte är att ersätta trafikljus eller att användaren inte ska titta på trafikljus för att undvika att passera när det är rött. Användaren ska justera sin hastighet för att uppleva en grön våg, men ska fortfarande följa trafikljusen så att hen inte åker rakt in i passerande trafik. Målet är att totala lyckade passeringar ska vara 100% med endast applikationen som hjälp, men 85.7% visar fortfarande att VeloFlow lyckas guida användaren, och med fortsatt justering av applikationen skulle den kunna ge ett värde ännu närmare 100%. Observationer från testerna pekar på att anledningen till att inte alla passeringar var lyckade var pågrund av att testpersonen inte följde instruktionerna som VeloFlow gav till fullo. Särkilt när testpersonen skulle sänka sin hastighet för att inte åka in i ett rött ljus. Användaren behöver sänka hastigheten tills de får ljudmeddelandet som signalerar att hastigheten är uppnådd. Detta verkade för vissa testpersoner ibland vara en allt för långsam hastighet, och de lyckades därför inte komma ner i tillräckligt låg hastighet för att lyckas med passeringen.



(a) Hur enkelt det var för testpersonerna att förstå instruktionerna som applikationen gav dem. Från 1 (väldigt svårt) till 5 (väldigt enkelt)

(b) Hur enkelt det var för testpersonerna att utföra instruktionerna som applikationen gav dem.

Figur 5.4: Resultat från slutanvändartest angående instruktionsförståelse och utförande

Något sammanband mellan cykelvana och antal lyckade passeringar kunde inte märkas av, inte heller andra faktorer som kön eller ålder hade någon märkbar påverkan.

5.2.3 Enkelt att förstå instruktioner

Efter att testpersonerna cyklat så fick de svara på några frågor, där två var att de skulle ge betyg på två faktorer. Dessa faktorer var hur enkelt det var att förstå applikationens instruktioner och hur enkelt det var att utföra dessa instruktioner. En sammanställning av svaren kan ses i diagrammen (a) och (b) i figur 5.4.

Svaren visar att det för dem flesta var enkelt att förstå vad VeloFlow ville att de skulle göra, men att det fanns utrymme att bli ännu tydligare. För att motivera sitt svar kunde testpersonerna berätta om en situation där de kände sig osäkra på vad de skulle göra medan de använde applikationen. De flesta nämnde då att VeloFlow i början eller senare under användning kunde ge många instruktioner med korta intervall, ibland pratade den till och med över sig själv. Då blev det svårt att veta vad applikationen ville att de skulle göra. Detta kan åtgärdas genom att introducera pauser mellan instruktioner som applikationen måste ta hänsyn till. En testperson tyckte att det var svårt att avgöra vad den skulle göra efter att den passerat ett simulerat trafikljus. Applikationen ger inget meddelande när användaren passerat ett trafikljus, utan låter användaren åka vidare i vilken hastighet den vill tills VeloFlow börjar ge nya instruktioner för nästa trafikljus. Detta framgick inte för alla under testning och visar ett tolkningsproblem som skulle behöva tas hand om. Ytterligare en kommentar från en annan testperson var att denna inte alltid hörde ljudmeddelandet från klockan, till exempelvis när det blåste. Detta ställde till det under testning men bör inte vara ett problem för användare som kan använda

hörlurar, som är det tilltänkta sättet att använda VeloFlow. Under testning användes smartklockans egna högtalare för att göra testerna mer hygieniska.

5.2.4 Förslag för att minska distraktion

Arbetet ”Co-riding With My eBike to Get Green Lights” [9] förklarar att för att användaren och systemets färdigheter ska kunna integreras så behövs koncisa kontextuella ledtrådar som skickas från systemet. Om dessa ledtrådar är för distraherande eller störande kan det anses att denna designtaktik inte uppnås och att samarbetet inte blir så bra som det skulle kunna vara mellan användare och system. Två av fem testpersoner upplevde ingen distraktion från att använda VeloFlow. Två upplevde viss distraktion ibland och en person rapporterade att applikationen var så pass distraherande att denne nog inte hade valt att använda applikationen om den fanns tillgänglig. Denna person tyckte att både ljudet och vibrationen som klockan spelade upp var för distraherande och irriterande, men mest ljudet var det som störde. En av de två som var måttligt distraherade påpekade också att ljudmeddelandena var de som var distraherande de gånger som det spelades upp för många tätt intill varandra.

Ytterligare en kommentar från användare var att det det skulle vara bra om det fanns möjlighet att välja vilken typ av ljudsignal som används, mellan ljudmeddelanden (röst med instruktioner) och ljudeffekter. Fördelar som lyftes för ljudmeddelanden var att det är tydligt vad som förväntas att användaren ska göra. Samtidigt kommenterade vissa användare att de föredrar ljudeffekter då dessa kan ta kortare tid att spela upp än ljudmeddelanden, vilket användare som normalt brukar lyssna på musik medan de cyklar anser mindre störande och distraherande.

5.2.5 Sammanfattning av slutanvändartesterna

I slutet av användartestet fick testpersonerna säga om de skulle kunna tänka sig att använda VeloFlow på riktigt om den lanserades med uppkoppling till riktiga trafikljus. Fyra av fem personer sa att de hade använt applikationen. Den femte personen sa att den inte hade använt applikationen för den vill ha lugn och ro när den är ute och cyklar, och hade därför stört sig på VeloFlow. Att kommunicera med denna användare hade därför krävt andra medel som inte skulle uppfattas som störande.

Sammanfattningsvis så gick testerna bra och de genererade användbar data som pekade starkt på att VeloFlow lyckades guida användarna utan större problem och att konceptet har ben att stå på. Vidare utveckling skulle kunna generera en applikation som kan hjälpa cyklister att få en grön våg, och då potentiellt få fler att börja cykla.

5.3 Batteriförbrukning

Mätningar av batteriförbrukning gjordes för när VeloFlow kördes och även när den inte kördes och smartklockan inte utförde mer än vanliga bakgrundsprocesser så som att möta puls och steg och skicka notifikationer. Mätningarna gjordes flera gånger med varierande längd, och ett medelvärde av procent förbrukat per minut räknades sedan ut. Batteriförbrukningen medan applikationen kördes blev i genomsnitt 0,456 procent per minut. I jämförelse förbrukas i genomsnitt 0,054 procent per minut när klockan inte används aktivt.

Detta resultat av batteriförbrukningsräkningen visar att när applikationen är igång och körs så dras 7.44 gånger mer batteri än när applikationen inte är igång och klockan endast körs passivt. Med passivt igång menas att klockan inte aktivt laddas utan sitter på en persons handled och mäter data som steg och puls och tar emot notifikationer. Detta kan ses som en extremt stor batteripåverkan, och hade VeloFlow varit igång en längre tid hade klockans batteritid försämrats avsevärt. Det som ska tas i åtanke är att applikationen endast ämnas att vara igång medan användaren cyklar. Om en användare pendlar med cykel sammanlagt 60 minuter på en dag så hade detta resulterat i en cirka 27.36 procent batteriminskning. Detta är märkbart men inte tillräckligt för att göra användingen av applikationen omöjlig. Det bör fortfarande gå att få ut en dagsanvändning utav klockan.

5.4 Designfaktorer för att skapa en grön våg

Resultaten från slutanvändartesterna och lärdomar från utvecklingen av systemet har genererat ny kunskap för detta forskningområde. Detta forskningsområde gällande smartklockeapplikationer som ska ge cyklister en grön våg kan antas vara litet och relativt outforskat utifrån den informationssökning som gjort under arbetets gång, och de lärdomar som erhållits är därför av stort värde för framtida arbeten inom detta område och andra områden som innehåller samma eller liknande faktorer. Dessa faktorer inkluderar men är inte begränsad till utveckling av smartklockeapplikationer, utveckling för applikationer riktade till cyklister, strategi för att uppnå grönvåg för cyklister och utveckling av molntjänst för hantering av trafikljusdata.

Denna nyfunna information har här sammanfattats till designfaktorer. Med designfaktor menas här en regel eller lärdom som bör följas och ta lärdom från i arbeten som relaterar till detta arbete eller delar av detta arbete.

5.4.1 Multimodal interaktion utan att distrahera cyklisten

Under projektets gång har en viktig del varit att undersöka olika sätt som det går att använda en smartklocka för att interagera på ett sätt som inte tar fokus från vägen för användaren. Från användartesterna i avsnitt 4.1 så framkom det att det för många var svårt att släppa en hand från styret då det påverkade balansen. Hur svårt det upplevdes att släppa en hand från styret berodde även på vilken uppgift det var som skulle utföras där vissa rörelser kändes mer naturliga än andra och

därför enklare. Att lyfta handen från styret för att läsa information på klockan upplevdes enklare än att lyfta en hand för att fysiskt klicka på en knapp eller svepa på pekskärmen. Finns det möjlighet att helt undvika att användaren behöver släppa händerna från styret för att använda applikationen så är det bäst.

Om information ska visas till användaren samtidigt som den cyklar så är det viktigt att tänka på hur denna information visas. Från användartesterna i avsnitt 4.1 så går det att konstatera att det för användarna var enklare att ta till sig information så som tid än att läsa längre textmeddelanden. Så för att göra det enklare för användaren bör information visas på ett visuellt och enkelt sätt så att användaren inte behöver lägga någon längre tid på att ta till sig informationen.

För att förmedla information på ett icke distraherande men tydligt sätt så är ljudsignaler en bra metod. Under användartesterna i avsnitt 4.1 var detta det sätt som flest användare tyckte enkelt att ta till sig, det var även en av de minst distraherande metoderna att ta in information. Vibrationer upplevdes dock lite mindre distraherande än ljudsignaler men med nackdelen att det var svårt för många användare att känna skillnad på olika vibrationer och i vissa fall även svårt att känna av vibrationerna över huvud taget medan de cyklade.

Under de avslutande användarstudierna (se avsnitt 5.2), så utvärderades VeloFlow och därmed också de olika sätten som applikationen förmedlade information till användarna. Bland kommentarerna som lämnades så var en att det var distraherande och svårt att förstå när pauserna mellan ljudmeddelandena för korta. Så för att minska distraktionen så kan begränsningar på hur ofta ny information skickas till användaren sättas upp.

Den designfaktor som här presenteras, hur en applikation för cyklister ska utvecklas till att vara distraktionsfri, kan sammanfattas till att ljudmeddelanden och vibrationer båda kan användas och ge bra resultat men att ingen av dem är problemfria. Denna designfaktor bygger vidare på det som ”Co-riding With My eBike to Get Green Lights” [9] påbörjade med sin designtaktik om kontextuella ledtrådar, men utökade kunskapen om vibration då detta inte är något som utforskades i deras arbete. Både vibration och ljud kan ses som alternativ för att kommunicera med cyklister, men fortsatt forskning skulle kunna tydligöra vibrationens möjligheter och begränsningar.

5.4.2 Design för olika användares preferenser

Att designa en applikation för alla användare är en utmaning då olika användare har olika preferenser. Bara för att något passar vissa användare så betyder det inte att det passar alla. Därför är det viktigt att användare ska ha viss valmöjlighet.

Ur resultatet av slutanvändartesterna, i avsnitt 5.2, så framkom det att vissa användare upplevde att ljudsignalerna var överflödiga och att det enligt dessa användare hade räckt med enbart vibrationer. Detta trots att det i de inledande användartes-

terna i avsnitt 4.1 framgått att det för många var svårt att känna av vibrationer. För de användarna som hade svårt att känna av vibrationerna så upplevdes vibrationerna i stället av vissa som distraherande då de behövde lägga fokus på att känna av vilken typ av vibrationer det var och för dessa användarna så hade det fungerat bättre att bara ha ljudsignaler.

Ytterligare ett exempel på preferens går att se på vilken typ av ljudsignal som föredras. Under användarstudier kring olika typer av ljudsignaler , i avsnitt 4.1, så fanns det ett konsensus om att ljudmeddelanden (upplästa meddelanden) var tydligare och enklare att förstå. Men trots detta så var det ändå en individ som under de avslutande användarstudierna lyfte önskemål om att kunna välja att ha ljudeffekter i stället för ljudmeddelanden. Detta då denna användaren ville ha något som inte blev ett lika stort avbrott i musiken som användaren ofta lyssnar på när den cyklar.

En annan viktig aspekt som bör tas hänsyn till är vilka språk de tilltänkta användarna är bekväma med. Att ge användare möjligheten att välja ett bekant språk gör det troligare att de faktiskt kan utnyttja applikationen till fullo. För detta projekt så har applikationen och ljudmeddelandena utvecklats för språken engelska och svenska då minst ett av dessa språk fungerade för alla användare som testade applikationen.

Dessa observationer och resultat kan te sig uppenbara men utifrån de svar som användare gett under användartesterna så är det tydligt att det finns en vinst i ge användaren valmöjligheter. Att erbjuda olika inställningar i applikationer hjälper då till att skapa en bättre användarupplevelse för varje individ.

5.4.3 Enkel prototyp som verktyg för att få in användarkrav

Inför de första användartesterna, i avsnitt 4.1, som utfördes för att få in användarkrav så hade ingen utveckling av applikationen skett. För att få in information om hur applikationen skulle designas användes en enkel prototyp. Denna enkla prototyp krävde ingen utveckling utan den testade olika scenarion genom att utnyttja inbyggd funktionalitet i de Garmin-klockor som användes.

För att testa hur användarna upplevde olika typer av vibrationer så utnyttjades det faktum att Garmin-klockorna hade en repeterad lång vibration vid inkommande samtal och korta vibrationer varje gång ett textmeddelande togs emot. Så för att testa olika mönster och hur väl användarna kunde känna skillnaden på olika typer av vibrationer så fick en av testledarna ringa eller skicka medelanden till den telefonen som var kopplad till smartklockan för att på så sätt utföra användartestet. Ljudsignaler testades på ett liknande sätt, där testpersonen fick ha på sig trådlösa hörlurar som en av testledarna kunde spela upp ljud på. De visuella delarna av användartestet lät användaren kolla på information som normalt sett visas på klockan, där ett test var att läsa av tiden och ett annat var att läsa en kortare text som kom i en notis.

Den simpla prototypen som inte krävde något förarbete möjliggjorde användarstudier i starten av produktutvecklingen vilket resulterade i att det redan i ett tidigt skede av projektet gick att sålla bort vissa idéer. Detta ledde till att tid inte slösades på utveckling av funktionalitet som ändå inte hade gett användarna något värde och att det gick att lägga mer fokus på de idéer som användarna kände hade potential.

5.4.4 Vikten av användarcentrerad design

Att utveckla för Wear OS i Compose ramverket för att ge cyklister en grön våg är ett område där väldigt lite forskning har publicerats. Det var därför svårt att veta vad som är extra viktigt för användarna av dessa system. Att arbeta med användarcentrerad design och ständigt samla in användarkrav och testa prototyper mot potentiella användare var därför väsentligt och gav mycket information för detta arbete.

Som förklarat i avsnitt 2.1, så krävs frekvent användartestning för en lyckad designprocess. Det är just detta som har fungerat för det här arbetet och som bör efterföljas för framtida arbeten där system riktar sig till smartklockor och cyklister. I början av arbetet hade vi förutfattade idéer om hur applikationen bör fungera, till exempel hur cyklisten ska få till sig instruktionerna från applikationen. Vi antog att cyklister skulle kunna känna av och urskilja olika vibrationer utan problem och att det skulle räcka med att ha små distinkta skillnader mellan vibrationsmönstrena. Med hjälp av användartestning, så insåg vi tidigt i arbetet att detta inte hade fungerat så bra för användaren. Istället för att upptäcka detta senare i utvecklingen efter att tid skulle ha lagts ner på att bygga en lösning som endast förlitade sig på vibrationer så kunde istället andra idéer tidigt utforskas och evalueras.

Vid utveckling utav system som det faktiskt finns mer erfarenhet inom krävs det ändå ett lika stort fokus på användarcentrerad design, men pågrund av tidigare erfarenheter kan förutbestämda idéer låsa in utvecklaren i en lösning som denne tror är optimal men som egentligen inte passar användaren alls på flera plan. Meningen med denna designfaktor är därför att förmedla att ett användarcentrerat-mindset är grundläggande för att fortsätta utvecklingen av en applikation som ska ge cyklister en grön våg och att de lärdomar som fåtts genom detta arbete angående användardesign bör tas som inspiration men inte ersätta framtida användartester för framtida projekt.

5.4.5 Val av cykelhastighet utifrån realistisk data

I arbetet har undersökningar gjorts om vilka faktorer som behöver användas för att göra VeloFlow så användbar som möjligt samtidigt som den tar hänsyn till en persons cykelförmåga. Den resulterande modellen som applikationen använder gör uträkningar baserade på nuvarande hastighet samt maximal hastighet för att avgöra hur mycket det går att accelerera och ifall användaren då kan hinna till ett trafikljus.

Initialt togs ingen hänsyn till cyklistens begränsningar i form av maximal hastighet

```
private fun distanceCalculationAccelerationCapped(
    timeToStatusChange : Double,
    accelerationConstant: Double,
    speed: Double
) : Double {
    var distance : Double
    if(speed+(accelerationConstant*timeToStatusChange) < maxSpeed){
        distance = speed*timeToStatusChange+(accelerationConstant*timeToStatusChange.pow( x: 2.0))/2.0
    }
    else{
        val timeBreakPoint = ((maxSpeed - speed)/accelerationConstant)
        distance = speed*timeBreakPoint+(accelerationConstant*timeBreakPoint.pow( x: 2.0))/2.0
        distance += maxSpeed * (timeToStatusChange - timeBreakPoint)
    }
    return distance
}
```

Figur 5.5: Kod för uträkning av maximalt täckt avstånd under en tid givet känd acceleration och hastighet.

och möjlighet att accelerera, men detta var något som identifierades som ett problem tidigt i utvecklingsfasen när en prototyp testades. Under dessa tester så var det ofta omöjligt att hinna komma upp i den hastighet som krävdes för att hinna med det gröna ljuset. Lösningen för detta problem blev att introducera en maximal cykelhastighet och en accelerationsfunktion in i modellen för att applikationen inte ska uppmana användaren att accelerera vid tillfällen som det inte är möjligt att hinna fram i tid till trafikljuset. Den maximala cykelhastigheten är den hastighet som användaren känner sig bekväm med att cykla i och vet att den klarar av att cykla i. Accelerationsfunktionen är i nuvarande implementation en stegfunktion baserat på data från accelerationstesterna från avsnitt 4.2.3. Utifrån accelerationsfunktionen, nuvarande hastighet, maximala hastigheten och tiden tills ett trafikljus byter status beräknar modellen om användaren kommer hinna fram i tid. Detta görs genom att teoretiskt räkna ut det maximala avståndet som cyklisten kan ta sig på den tid det är kvar innan trafikljuset slår om. Beräkningen av avståndet sker med hjälp av koden i figur 5.5 och jämförs sedan med cyklistens avstånd från trafikljuset för att se om cyklisten har möjlighet att hinna fram innan det slår om till rött. Ifall det beräknade avståndet cyklisten kan ta sig på tiden innan trafikljuset slår om till rött är större än avståndet cyklisten har kvar till trafikljuset så kommer applikationen då att ge rekommendationen att öka hastigheten.

Ytterligare ett problem upptäcktes, vilket hade att göra med det avstånd som cyklisten skulle ha för att få instruktioner för ett kommande trafikljus. Funktionen gick ut på att utifrån cyklistens nuvarande hastighet ge instruktioner på olika avstånd från trafikljus. Till exempel gavs instruktioner på ett avstånd av 75 meter om cyklisten hade en hastighet som var lägre än 4 m/s (≈14.4 km/h), om hastigheten däremot var mellan 4 m/s (≈14.4 km/h) och 6 m/s (≈21.6 km/h) var motsvarande avstånd 150 meter och om cyklisten var snabbare än 6 m/s (≈21.6 km/h) var avståndet 225 meter. Instruktionerna om vilken handling som skulle utföras kom i den första iterationen utav denna uträkning på för korta avstånd från trafikljuset baserat på

den hastighet som hölls, vilket gjorde att det ofta blev för lite tid att reagera på instruktionerna vid högre hastigheter. Lösningen blev helt sonika att öka avstånden till de som nu står i denna förklaring.

Dessa två uträkningar bör stå som grund för framtida utvecklingar utav modellen för denna applikation och framtida varianter eftersom de har testats och givit bra resultat. På denna logiska modell kan fortsatt arbete med förbättringar och utveckling göras där tanken bakom uträkningarnas urspung, att ta hänsyn till cyklisters fysiska förmåga, bör stå i centrum vid framtida arbeten.

5.4.6 Användning av molntjänst för att särkoppla data

Under projektets gång så har en molntjänst, AWS, använts för att simulera trafikljus och utföra beräkningar kopplade till trafikljusen. Genom att flytta ansvaret från smartklockeapplikationen till molnet så separeras ansvar och förutsättningar skapas för att smidigare kunna uppdatera delar av logiken efter hand. Så länge API:t för kommunikation mellan smartklockeapplikationen och molnet inte ändras så går det helt fristående att utveckla både molntjänsten och smartklockeapplikationen utan att det påverkar den andra kodbasen. Att undvika att behöva uppdatera smartklockeapplikationen är något som är önskevärt, då det inte ställer lika höga krav på användaren att uppdatera applikationen så ofta.

Att molntjänsten är ansvarig för att sköta all logik kopplad till trafikljusen innebär att det går att uppdatera data för molntjänsten genom att till exempel lägga till fler trafikljus eller att byta ut algoritmen för avståndsberäkning utan att någon uppdatering behöver göras på smartklockeapplikationen. Det innebär också att det enklare skulle gå att koppla systemet mot faktiska trafikljus då det räcker att ändra datakälla för molntjänsten. Att detta är möjligt beror på att molntjänsten automatiskt kan skalas upp för att hantera de beräkningar som behövs.

Molntjänsten möjliggjorde även att det enkelt under projektets gång gick att utveckla stödapplikationen i avsnitt 4.5. Den information som stödapplikationen behövde var status på trafikljusen och hur lång tid det var kvar tills status skulle ändras, vilket skiljer sig från den information som smartklockeapplikationen behöver. Tack vare molntjänstens flexibilitet gick det enkelt att modifiera koden i molnet så att både stödapplikationen och smartklockeinformationen kunde få den information som de behövde, baserat på hur molntjänsten anropades, utan att öka komplexiteten.

Under projektets gång har användandet av molntjänsten AWS underlättat delar av produktutvecklingen då det har brutit upp projektet i mindre delar som går att utveckla parallellt. Det har även gjort att framtida förändringar enklare kan genomföras. För liknande arbeten inom liknande scenarion är en rekommendation att utnyttja de fördelarna som en molntjänst kan erbjuda för att öka kvalitén på produkten samtidigt som utvecklingsprocessen blir enklare då problemet bryts ner.

6

Diskussion

I detta kapitel diskuteras olika aspekter angående arbetet, så som dess påverkan på omvärlden, hur bra de valda metoderna fungerade, inkluderande design och framtida utveckling av VeloFlow.

6.1 VeloFlow

Prototypen VeloFlow visar att konceptet för en grön våg-applikation kan fungera. Detta arbete kan sedan leda till fortsatt arbete inom området och potentiellt leda till utvecklingen av en applikation som går att använda i den verkliga trafiken. Det är här som de designfaktorer som presenterats kan hjälpa till. Genom att följa och ta lärdom från dem kan framtida arbeten bygga vidare på designfaktorerna. När de används kan utvecklaren försäkra sig om att de testade designfaktorerna kommer att förbättra deras arbete.

6.1.1 VeloFlows samhällspåverkan

Om VeloFlow utvecklas till en fullskalig applikation som fungerar med riktiga trafikljus skulle fler trafikanter kunna välja bort bilen och istället ta cykeln till jobbet och andra platser. Som förklarat i bakgrunden, så skulle detta kunna bidraga till att årligen rädda 449 år av livslängd i Stockholm[5]. Det hade även minskat nivån som biltrafik påverkar den globala uppvärmingen[6]. De framsteg som gjorts i detta arbete kan därför ha en direkt positiv inverkan på klimatkrisen som råder i världen och rädda liv i Sverige, potentiellt även över hela världen. Arbetet följer därför en hållbar utveckling på både ett ekologisk och socialt vis.

För att få folk att börja använda VeloFlow och då börja cykla mer så krävs det att användarna känner sig trygga. Mycket arbete har gjorts för att minimera risken att skada sig, men det finns fler säkerhetsaspekter som behöver tas i åtanke. För att VeloFlow ska fungera krävs det att användarens exakta position alltid är känd, och detta kan i händelsen av en cyberattack utgöra ett integretetsproblem. Potentiell data så som cykelvana och vanliga platser användaren besöker, så som hem och arbete, är också känslig data som kan samlas in för skadligt bruk om fel händer fick tag på den. Det är inte socialt hållbart att erbjuda tjänster som ska gynna samhället

om de inte kan erbjudas på ett säkert sätt. Skulle VeloFlow och dess molntjänst utvecklas vidare till en fullskalig tjänst hade det därför krävts att åtgärder togs för att försäkra att användarens data är säker.

VeloFlow kan hjälpa Göteborg stad och andra städer att förbättra livet för cyklister och potentiella cyklister på ett ekonomisk hållbart sätt, eftersom VeloFlow inte i sig kräver att infrastrukturen byggs om. Ombyggnation kan möjliggöra ännu bättre cykling i städer och är därför också en möjlig lösning för att öka cykling och minska bilkörandet, men i direkt jämförelse är VeloFlow ett ekonomiskt mer hållbart sätt som därför kan användas parallellt med ombyggnationer.

6.1.2 Framtida arbete

I detta avsnitt lyfts potentiella förbättringar för VeloFlow som inte har hunnits med under projektets gång eller som av andra anledningar inte rymts inom ramen för detta examensarbete.

6.1.2.1 Ruttplanering

De simulerade trafikljusen har alla en gemensam egenskap: de är alla individuellt ensamma vid sin simulerade korsning. Vid testning av VeloFlow så var alltså alla trafikljus ensamma vid sina punkter, men på verkliga vägar är det inte ovanligt med korsningar där det finns flera trafikljus vid samma punkt som korsar vägar i olika riktningar. Dessa trafikljus delar sällan samma status, så det krävs att VeloFlow ska veta vilket trafikljus som cyklisten tänkt passera för att applikationen ska kunna skicka korrekta instruktioner. Alternativet är att skicka information till cyklisten om alla trafikljus vid en korsning, men den informationen hade snabbt blivit överväldigande och förvirrande för cyklisten, och hade möjligtvis krävt att cyklisten tittade mer aktivt på klockans skärm. Det bör som tidigare förklarat helt undvikas på grund av den risk för olyckor som ökar när cyklister blir distraherade.

För att veta vilket trafikljus som cyklisten ska information om krävs alltså att VeloFlow vet om den rutt som cyklisten ska åka, och det skulle kunna delges genom att användaren innan påbörjad cykeltur skriver in sin slutdestination genom en navigationstjänst, som exempelvis Google Maps. Navigationstjänsten kan då stå för ruttplaneringen, och sedan skicka denna information till vår applikation och molntjänst som då vet vilka trafikljus den borde sikta in sig på.

En extra funktion som går att vinna på den här integreringen av ruttplanering är att vår applikation kan skicka tillbaka information om trafikljus till ruttplaneraren som då kan planera om en rutt beroende på trafikljusens status. Om användaren kommer fram till en korsning där den snabbaste vägen i vanliga fall är att passera trafikljuset rakt fram, men pågrund av trafikljusens status så är det nu snabbare att passera trafikljuset till vänster, så kan applikationen förmedla detta till ruttplaneraren. Ruttplaneraren ändrar då på den planerade rutten och kommunicerar denna ändring till användaren som kan agera på det. På det sättet effektiviseras användarens cykling ännu mer och får ytterligare ett verktyg för att slippa sakta

ner pågrund av rött ljus.

6.1.2.2 Personalisering av applikationen

En punkt som går att utveckla för att göra VeloFlow bättre för fler användare är att ge användare möjlighet att anpassa applikationen efter sin egna cyklingsförmåga. I modelen som används för att ge rekommendationer för vad användaren ska göra används antaganden om maxhastighet och möjlig acceleration. Genom att ge användaren möjlighet att ange sin cykelvana, hur snabbt de normalt brukar cykla och deras uppskattade maxhastighet så hade det gått att personalisera applikationen så att den ger bättre rekommendationer. Till exempel kan VeloFlow i nuläget rekommendera en användare att accelerera för att applikationen tror att cyklisten kan cykla snabbare än vad den kan, vilket kan leda till att användaren börjar accelerera även om den inte har möjlighet att uppnå hastigheten som krävs för att hinna i tid. Detta kan undvikas genom att anpassa applikationsmodellen till att beräkna rekommendationerna utifrån faktiska värden för varje användare.

6.1.2.3 När cyklisten inte vet avståndet till trafikljuset

Ett förslag som gavs under slutanvändartesterna och presenterades i resultatet var att användaren skulle kunna få information om när denne börjar närma sig ett eller flera trafikljus. Så som VeloFlow fungerar nu så ges ingen information om hur nära användaren är det trafikljus som den får instruktioner för. Om användaren cyklar en okänd väg så kan det möjligtvis bli svårt att avgöra om användaren orkar att hålla en snabbare takt då den inte vet hur lång distans den behöver göra det. Kanske hade det istället passat bättre att sakta ner och låta applikationen ge instruktioner för det. VeloFlow är ett stöd och fungerar allra bäst när användaren kan vara med och bestämma om den ska höja eller sänka farten genom att avväga informationen den får från applikationen och informationen den bär med sig om omgivningen. En framtida utveckling skulle därför kunna vara att förmedla mer information till användaren angående distans till nästa trafikljus, och detta skulle kunna kombineras med instruktionerna som användaren skulle få om ruttplaneringen hade implementerats i VeloFlow.

6.1.2.4 Hälsodata

Smartklockor har vanligtvis pulsmätare installerade, och dessa skulle kunna användas för att se över cyklistens status medan VeloFlow körs. Smartklockor har också själva olika applikationer för att mäta hälsodata under aktiviteter, men istället för att behöva starta både en av de applikationerna och vår cykelapplikation så kan VeloFlow sköta båda delar. När användaren sedan är klar med sin cykling och avslutar applikationen kan användaren få del av sin hälsodata och se den utveckling som sker genom att använda VeloFlow för exempelvis sin jobbpendling. Detta hade givit ännu ett incitament att använda applikationen och att ta cykel istället för bilen.

6.1.2.5 Få information om sin klimatpåverkan

En utvecklingspunkt som diskuterades under arbetet var att ge användaren data angående hur den gjort en positiv påverkan på klimatet genom att välja cykel istället för bilen. Detta hade kunnat ge ännu mer incitament till att välja det miljövänliga transportmedlet. Tyvärr fanns det inte tid till att implementera detta under detta arbetets gång, men en implementation av detta som det skapades vissa planer för var att när användaren avslutade sin cykling så skulle applikationen visa hur mycket koldioxid användaren sparat genom att inte ta bilen. Applikationen hade endast behövt se över den sträcka som användaren cyklat för att se hur mycket koldioxid som hade släppts ut med den genomsnittliga bilen.

6.1.2.6 Utveckla applikationen för fler operativsystem

Under detta projektets gång har fokus legat på att uteveckla en applikation för enheter med operativsystemet Wear OS. Efter utvecklingen av applikationen för Wear OS har det konstaterat att VeloFlow har potential att fungera även på smartphones. Hälsodata så som puls hade förlorats om användaren körde applikationen på sin smartphone, men eftersom grundfunktionaliteten av att få information igenom ljudmeddelanden och vibrationer fortfarande går att få via smartphone så hade VeloFlow kunnat uppnå sin huvuduppgift. Så en utvecklingspunkt är att göra en variant av applikationen för Android-telefoner. Det finns även en vinst i att utveckla varianter av applikationen mot Apples operativsystem iOS och watchOS så att även IPhones och Apple watch kan användas för att köra VeloFlow. Att utveckla mot fler operativsystem gör att det är fler som får tillgång till applikationen utan att behöva investera i ny hårdvara.

6.1.2.7 Fortsatt testning av batteriförbrukning

För att veta exakt vilken del klockan som står för majoriteten av batterikonsumptionen hade det krävs ytterligare testning där olika delar av applikationen avaktiverades och mätningar av batteriet utfördes. Utan att lägga den extra tiden så kan en säker gissning vara att det som kräver mycket batteri är den frekventa uppdateringen av platsdata. För att applikationen ska ha hög precision hämtas så exakt platsdata det går att få tag på genom klockans system flera gånger i sekunden. Den gör detta för att alltid ha korrekt data att skicka till molntjänsten så att den i sin tur ha så korrekt data som möjligt att skicka tillbaka till applikationen. De starka vibrationerna som används kan också ha en märkbar batteripåverkan.

Vid vidareutveckling av applikationen hade tid lagts på att förbättra batteriförbrukningen, och möjligtvis även testa förbrukningen över fler modeller av smartklockor då detta också kan ha stor påverkan.

6.1.2.8 Koppling av applikationen till riktiga trafikljus

Under starten av detta projekt så initierades kontakt med Stadsmiljöförvaltningen i Göteborgs stad för att se över möjligheten att koppla applikationen till riktig data från trafikljus. Detta var inte möjligt att göra under projektets gång då de

inte hade utvecklat klart lösningen som krävs för att koppla externa tjänster till trafikljusinformation som tidigare förklarats i avsnitt 4.3.

För att VeloFlow ska vara användbar i verkliga scenarion så krävs det att arbetet med att koppla upp VeloFlow mot riktig data från trafikljus genomförs. Uppkopplingen mot trafikljus hade inneburit att en anpassning av molntjänsten behövs genomföras för att hämta data på ett effektivt sätt från trafikljusens informationssystem. För att anpassa molntjänsten så hade först API för trafikljussystemet undersökas för att se hur anrop till det behöver göras och vilken data som svar hade innehållit. Utifrån detta så hade molntjänsten behövt omarbetas till att använda den tillgängliga datan på ett effektivt sätt för att ge applikationen den information som behövs.

6.1.2.9 Utveckling för att anmäla gröntljusbehov

Under förutsättningen att applikationen och molntjänsten har en koppling till trafikljusinformation så hade det gått att effektivisera ytterligare för cyklister genom att anmäla gröntljusbehov digitalt via molntjänsten. Detta hade inneburit att en signal som signalerar att cyklister vill få grönt ljus för att få passera skickas till trafikljussystemet när en eller flera cyklister närmar sig ett trafikljus. Fördelen med att göra detta med hjälp av applikationen och molntjänsten är att det går att anmäla gröntljusbehov antingen baserat på avstånd till trafikljuset eller en viss tid innan cyklisten är beräknad att vara framme vid trafikljuset. Detta möjliggör att det går att optimera när det blir grönt bättre än dagens lösning där fysiska knappar framme vid trafikljuset och givare utsatta på ett visst avstånd från trafikljusen används för att anmäla gröntljusbehov.

6.2 Metodreflektioner

Här följer kortare diskussioner om vad som funkat bra och mindre bra med de valda metoderna för detta arbete, samt hur designen kan bli mer inkluderande.

6.2.1 Scrum

Som förklarat i avsnitt 2.3, så utfördes detta arbete enligt det agila ramverket Scrum för att förenkla arbetsflödet och göra det tydligt vad som skulle göras och vem som skulle göra det. Att arbeta agilt med Scrum har fungerat väldigt väl för detta projekt. Frekventa möten med produktägarna, som är handledarna på Knightec, erbjöd snabb respons på nya funktioner och förändringar i utvecklingsplanen kunde göras tidigt för att undvika att arbete slängs längre fram. Sprintarna var av bra längd och dagliga möten missades nästan aldrig, vilket bidrog till att båda utvecklarna i teamet alltid visste om vad den andra gjorde och båda kände att de kunde påverka systemets utveckling trots att arbetsuppgifter var fördelade mellan dem.

Det som skulle kunna ha gjorts bättre under arbetets gång var att specificera arbetsuppgifter. Under arbetet användes en så kallad ”Scrum board” där uppgifter

sattes upp och skapade en överskådlig bild av det som skulle göras. Denna sorts anslagstavla började användas mer flitigt mot slutet av arbetet, men det hade hjälpt till att tydliggöra vad som behövde göras för utvecklare och produktägare. Det hade i sin tur lett till mindre energi lagd på att hålla reda på arbetsuppgifter i minnet.

6.2.2 Avvikelse från MVVM

I utvecklingen för Wear OS-applikationen så följdes designmönstret MVVM så bra som möjligt, men den resulterande applikationen avviker något från mönstret. Detta beror delvis på att det inte fanns många exempel på hur MVVM ska följas för en Wear OS-applikation som använder ramverket Jetpack Compose. Applikationen förenklades ner till en simpel struktur som förlitade sig på views som använde en grupp utav verktyg. Viewmodels blev i denna design en överflödig abstrahering som endast hade gjort strukturen otydligare. Den resulterande strukturen var enkel att arbeta med och göra förändringar i. Att börja använda MVVM som bas gjorde applikationen lättmanövrerad, men att våga gå iväg från mönstret där det var nödvändigt gjorde att slutprodukten enklare kunde hantera de nödvändiga verktygen, så som Androids foreground service (se avsnitt 4.2.1 för förklaring utav foreground service).

6.2.3 Datainsamling från användare

De metoder som använts för att samla in data för användarkrav och evaluering har fungerat bra men har även haft vissa svagheter. Metodernas styrkor och svagheter presenteras här i följande underavsnitt.

6.2.3.1 Strukturerade intervjuer

Intervjuerna som användes i detta arbete var av den strukturerade varianten, och de möjliggjorde enkel informationssamling utav både kvalitativ och kvantitativ data. De var enkla att skapa och utförandet var smidigt eftersom det fanns en fast plan som kunde följas för de frågor som skulle ställas. Det var också positivt att det gick ett förutspå ungefär hur lång tid varje intervju skulle ta eftersom de alla liknade varandra. Efter att första var utförd visste man därför hur lång tid det tog att svara på frågorna och eftersom inga ytterligare följdfrågor än dem som var planerade ställdes skiljde sig inte intervjuerna åt något större. Svaren från intervjuerna var dessutom enkla att grupper och sammanfatta eftersom de alla svarade på exakt samma frågor.

Hade inte strukturerade intervjuer använts så är det inte säkert att lika mycket data skulle ha hunnit samlas in och det är inte heller säkert att den hade varit av samma kvalité. Ostrukturerade intervjuer hade möjligtvis givit mer detaljerad data och kunnat utforska oväntade ämnen som först kommer på tal under en intervju och inte i planeringen inför den. Frågeformulär som skickas ut hade kunnat möjliggöra svar från en större grupp, men hade inte gett möjligheten att testa prototypen med personer på plats och på det sättet få mer sanningsenlig data. Det hade inter heller gjort det möjligt att utnyttja det andra verktyget för datainsamling: observationer.

6.2.3.2 Observationer

För att ge stöd till de strukturerade intervjuerna och fylla i informationshål som kan skapas genom att testpersonerna ibland ger falska tankar eller åsikter trots att de själva tror att de stämmer så användes observationer. Observationerna skrevs ner under testerna och kunna användas för att jämföra testpersonernas egna svar. Dessa observationer var mycket hjälpsamma när testpersonernas uppmärksamhet och förmåga att kontrollera cykeln de satt på testades. En testperson kunde ibland svara att det kändes stabilt men observationerna visade att det faktiskt inte stämde. I dessa fall hjälpte därför observationerna till med att den insamlade datan var tillförlitlig.

Observationerna var enkla att skriva ner och tog inte så lång tid att analysera, men det var i vissa fall svårt att strukturera upp deras format. Hur observationer skrevs ner kunde skilja sig mellan användartesterna och detta gjorde att de kunde variera i kvalité. Mer arbete kunde därför ha lagts ner på att bestämma en struktur för hur observationer skulle noteras. Ytterligare ett mindre problem som upptäcktes med observationer var det att de gör det svårare för testledare att faktiskt observera testerna. För att se till att observationerna var så korrekta som möjligt skrev de ner under testets gång istället för att behöva komma ihåg observationerna i minnet. Detta krävde ibland att personen som skrev ner observationerna tittade undan från testpersonen och då kunde missa detaljer. Trots detta så kom de viktigaste detaljerna med, främst genom att testledarna alltid var två och att minst en alltid tittade på den aktuella testpersonen.

6.2.4 Inkluderande design

Det går att argumentera för att en exkluderande design är rent av oetiskt om det exkluderar användare från detta arbetets applikation och då ger dessa personer sämre förutsättningar att få en grön våg än personer som kan använda applikationen. För att nå en större användargrupp, minska exkluderingen och förbättra rättvisan i vilka som kan utnyttja att få en grön våg vid cykling så hade vidare arbete behövts utföras för att göra applikationen mer inkluderande.

En användargrupp som riskerar att bli exkluderade är döva personer eller personer med hörselnedsättning. Dövhet stoppar inte en person från att kunna cykla och hörselnedsättning bör därför inte stå som grund för att inte få uppleva en grön våg vid cykling. Myndigheten för digital förvaltning (DIGG) presenterar på hemsidan om vägledning för webbutveckling en riktlinje som delger att inspelningar som endast ges i ljudform ska ackompanjeras med alternativa sätt att få till sig information i inspelningen[30]. Detta arbete har inte bestått utav webbutveckling, men deras riktlinjer kan fortfarande ses som relevanta för att skapa en inkluderande design oberoende på tekniskt medium. De ljudmeddelanden som VeloFlow använder sig av behöver enligt denna riktlinje dubbelkodas så att informationen går att få till sig på flera sätt. Applikationen har vibrationer som ska ge samma information som ljuden gör, men från testning har det framgått att vibrationerna inte är lika enkla att förstå och ofta är otillräckliga för att ensamt kunna instruera användaren. Fortsatt

arbete med vibrationerna för att göra dem tydligare eller utforskning i alternativa kommunkationsvägar hade därför behövts för att bättre kunna inkludera döva personer.

6.2.4.1 Inkludering genom vald metod

Eftersom detta arbete har följt UCD-principer så kan viss försäkran ges om att den mest typiska användaren kommer att kunna få ut värde ur VeloFlow om den kopplades upp till ett riktigt trafikljusnät. Att arbeta med UCD som mål innebär att testning och evaluering har gjort löpande under projektets gång och den tänkta användarens övergripande krav bör vara täckta. Vid fortsatt utveckling hade vi fortsatt att tillämpa UCD eftersom det har givit bra resultat.

7

Slutsats

De framsteg som gjorts under detta arbete kommer kunna stå som grund till framtida projekt inom utvecklingen av applikationer som ska ge cyklister en grön våg, och även andra applikationer för smartklockor som har i uppgift att kommunicera med cyklister. Designfaktorerna som presenterades i resultatet formulerades utifrån de lärdomar som arbetet gav. De ämnar att tillföra värde inom flera områden så som applikationer för cyklister och lösningar för grön våg samtidigt som de potentiellt kan bidra till att minska fossildrivna fordons påverkan på den globala uppvärmingen. Detta arbetet kan även hjälpa Göteborg stad att nå sina mål som ska göra Göteborg till en cykelvänligare stad. Med fortsatt utveckling utav VeloFlow kan cyklister på väg till jobbet, hemmet, skolan eller annan plats få en trevligare och smidigare cykelupplevelse. Vilket kan göra det mer troligt att fler väljer cykeln som transportmedel, som kommer att göra vår planet till en bättre plats att bo på.

Litteraturförteckning

[1] O. Tagesson, ”Göteborg är ute och cyklar”,*Naturskyddsföreningen*, [Online], 2022. Tillgänglig: https://goteborg.naturskyddsforeningen.se/vara-standpunkter/goteborg-ar-ute-och-cyklar/#lank-rubrik-3 (hämtad: 2023-04-09).

[2] J. Karlström, A. Niska, ”Cyklingens koppling till Agenda 2030: Systemtänkande i transportsektorn”, Statens väg- och transportforskningsinstitut, Stockholm, Sverige, VTI rapport 1130, 2022, [Online] Tillgänglig: https://www.diva-portal.org/smash/get/diva2:1691423/FULLTEXT01.pdf (hämtad 2023-05-09).

[3] Svenska FN-förbundet, ”Globala målen för hållbar utveckling”, 2023. [Online]. Tillgänglig: https://fn.se/globala-malen-for-hallbar-utveckling/ (hämtad 2023-05-09).

[4] Cykelfrämjandet, ”Cyklistvelometer 2022”, 2022. [Online]. Tillgänglig: https://cykelframjandet.se/cyklistvelometern/ (hämtad 2023-05-09).

[5] C. Johansson et al., *”Impacts on air pollution and health by changing commuting from car to bicycle”*, Science of The Total Environment, vol. 584-585. ss. 55-63, Apr, 2017, doi: https://doi.org/10.1016/j.scitotenv.2017.01.145 (hämtad 2023-01-23)

[6] E. Mathez, J. Smerdon, *”Climate Change: The Science of Global Warming and Our Energy Future”*, 2 uppl, New York, USA: Columbia University Press, 2018. [Online]. Tillgänglig: https://ebookcentral.proquest.com/lib/chalmers/detail.action?docID=5276366# (hämtad 2023-01-23)

[7] M. Månsson, M Junemo, *”Cykelprogram för en nära storstad 2015-2025”*, Trafikkontoret, Göteborg, Sverige, ISSN 1103-1530, 2015. [Online]. Tillgänglig: https://tekniskhandbok.goteborg.se/wp-content/uploads/1D_43_Cykelprogram-for-en-nara-storstad-2015-2025.pdf (hämtad 2023-01-20)

[8] K. Sandin, *”Svårt nå målen i Göteborgs Stads cykelpro-*

gram trots många insatser", Vårt Göteborg, Feb, 2, 2022. [Online]. Tillgänglig: https://vartgoteborg.se/trafik/svart-na-malen-i-goteborgs-stads-cykelprogram-trots-manga-insatser/ (hämtad 2023-01-23)

[9] J. Andres, T. Kari, J. Kaenel, F. Mueller, *"Co-riding With My eBike to Get Green Lights"*, i Proceedings of the 2019 on Designing Interactive Systems Conference (DIS '19), New York, USA, 2019, ss. 1251-1263. [Online]. Tillgänglig https://dl.acm.org/doi/10.1145/3322276.3322307 (hämtad 2023-01-27)

[10] A. Pratt, J. Nunes, *'Interactive Design : An Introduction to the Theory and Application of User-Centered Design"*, 1a uppl, USA: Quarto Publishing Group USA, 2012. [Online]. Tillgänglig: https://ebookcentral.proquest.com/lib/chalmers/reader.action?docID=3399638 (hämtad 2023-05-09).

[11] H. Sharp, J. Preece, Y. Rogers, *"Interaction Design: Beyond Human-Computer Interaction"*, 5e uppl, Indianapolis, USA: John Wiley & Sons, Incorporated, 2019. [Online]. Tillgänglig: https://search.ebscohost.com/login.aspx?direct=true&db=cat07472a&AN=clec.EBC5746446&site=eds-live&scope=site (hämtad 2023-02-07).

[12] Intigritetsskyddsmyndigheten, *"Grundläggande principer"*, 2021. [Online]. Tillgänglig: https://www.imy.se/verksamhet/dataskydd/det-har-galler-enligt-gdpr/grundlaggande-principer/ (hämtad 2023-02-09)

[13] M.Comstedt, *"Grunderna i Scrum"*, ONBIRD, 2018. [Online] Tillgänglig: https://onbird.se/grunderna-i-scrum/ (hämtad 2023-02-09)

[14] Android developers, *"Wear OS versus Mobile Development"*, Januari, 23, 2023. [Online]. Tillgänglig: https://developer.android.com/training/wearables/wear-v-mobile (hämtad 2023-05-04)

[15] Android developers, *"Why adopt Compose"*, Mars, 1, 2023. [Online]. Tillgänglig: https://developer.android.com/jetpack/compose/why-adopt (hämtad 2023-03-15)

[16] Android developers, *"Jetpack Compose Phases"*, Mars, 6, 2023. [Online]. Tillgänglig: https://developer.android.com/jetpack/compose/phases(hämtad 2023-05-04)

[17] Android developers, *"Thinking in Compose"*, Maj, 3, 2023. [Online]. Tillgänglig: https://developer.android.com/jetpack/compose/mental-model (hämtad 2023-05-04)

[18] D. Chang, *"Declerative vs imperative programming: 5 key differences"*,

Juli, 18, 2022. [Online]. Tillgänglig: https://www.educative.io/blog/declarative-vs-imperative-programming

[19] Android developers, *"Meet Android Studio"*, April, 24, 2023. [Online]. Tillgänglig: https://developer.android.com/studio/intro (hämtad 2023-05-03).

[20] Amazon Web Services, *"Cloud computing with AWS"*, 2023. [Online]. Tillgänglig: https://aws.amazon.com/what-is-aws/ (hämtad 2023-05-03).

[21] Amazon Web Services, *"What is AWS Lambda?"*, 2023. [Online]. Tillgänglig: https://docs.aws.amazon.com/lambda/latest/dg/welcome.html (hämtad 2023-05-03).

[22] Amazon Web Services, *"What is Amazon API Gateway?"*, 2023. [Online]. Tillgänglig: https://docs.aws.amazon.com/apigateway/latest/developerguide/welcome.html (hämtad 2023-05-02)

[23] Amazon Web Services, *"What is RESTful API?"*, 2023. [Online]. Tillgänglig: https://aws.amazon.com/what-is/restful-api/ (hämtad 2023-05-02).

[24] Next Bike, *"Styr & Ställ"*, 2023. [Online] Tillgänglig: https://styrochstall.se/sv/ (hämtad 2023-03-27)

[25] Google Play Services, *"FusedLocationProviderClient"*, Oktober, 13, 2022. [Online]. Tillgänglig: https://developers.google.com/android/reference/com/google/android/gms/location/FusedLocationProviderClient.html (hämtad 2023-04-05)

[26] Android developers, *"Principles of Wear OS development"*, Februari, 28, 2023. [Online]. Tillgänglig: https://developer.android.com/training/wearables/principles (hämtad 2023-04-05).

[27] Android developers, *"Foreground services"*, Mars, 29, 2023. [Online]. Tillgänglig: https://developer.android.com/guide/components/foreground-services (hämtad 2023-04-05).

[28] jsfxr, "8 bit sound maker and online sfx generator", [Online]. Tillgänglig: HTTPs://sfxr.me/ (hämtad 2023-06-01)

[29] FreeTTS, "Free TTS: Text to Speech Mp3 Free Online", [Online]. Tillgänglig: https://freetts.com/ (hämtad 2023-06-01)

[30] Myndigheten för digital förvaltning, "Erbjud alternativ om en inspelning enbart består av ljud eller video", 2018. [Online]. Tillgänglig: https://webbriktlinjer.se/riktlinjer/116-alternativ-vid-enbart-ljud-video/ (hämtad 2023-05-10)

A

Bilaga - Instruktioner för användartest

Instruktioner för användartest

•••

Idén med applikationen

- Att hjälpa dig komma i tid till trafikljuset
  - Detta innebär att du inte är för tidig eller för sen
- För att informera dig om vad du ska göra används ljudsignaler och vibrationer

Ljudsignaler

Öka hastigheten:

Sänka hastigheten:

Hastighet uppnådd

Vibrationer

- Kommer att vibrera när ni ska öka eller sänka hastigheten, för att veta vad så får man lyssna på ljudsignalen
- När rätt hastighet uppnåtts så slutar klockan att vibrera

Tänk på säkerheten!

Glöm inte bort din omgivning!

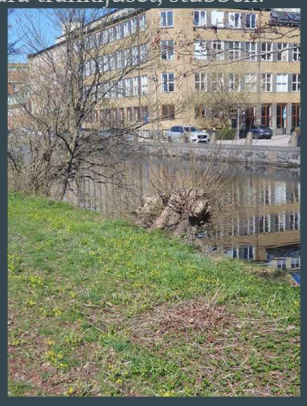
Avstå om du inte kan utföra höjning eller minskning av hastigheten på ett säkert sätt!

“Trafikljusen”

Det första trafikljuset, väjningspliktsskylten:

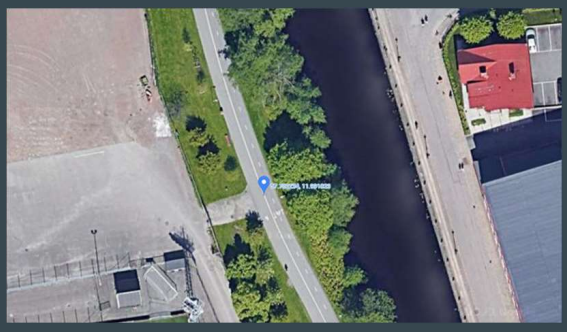
“Trafikljusen”

Det andra trafikljuset, stubben:



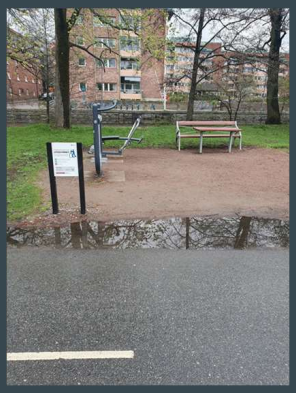
“Trafikljusen”

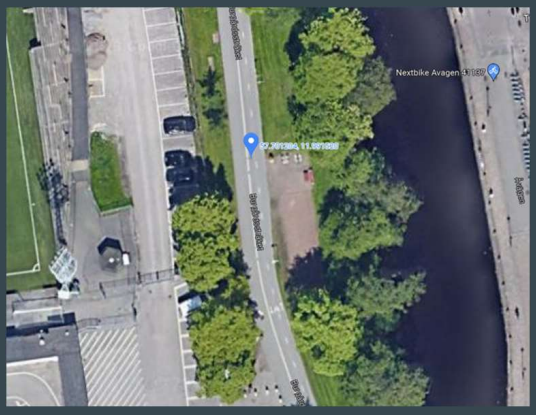
Det tredje trafikljuset, informationsskylten:



"Trafikljusen"

Det fjärde trafikljuset, utegymmet:





Vad ska jag göra?

- Signera medgivandeformulär för datahantering
- Cykla precis som du normalt hade gjort och försök lyssna på och följa instruktionerna från applikationen (om det är möjligt)
- Vi kommer under tiden att observera om du hinner med de gröna ljusen med hjälp av applikationen
- Efter du har cyklat kommer du få svara på hur du upplevde användningen och om det finns något du tycker borde förändras/förbättras

Sträcka att cykla

Från Ullevi till strax efter utegymmet vid Valhalla IP

Hela denna sträckan två gånger.

Sen två gånger på en kortare sträcka mellan Ullevi och korsningen vid Levgrensvägen

INSTITUTIONEN FÖR DATA OCH INFORMATIONSTEKNIK
CHALMERS TEKNISKA HÖGSKOLA
Göteborg, Sverige
www.chalmers.se

GÖTEBORGS
UNIVERSITET

CHALMERS