M.Sc. Thesis
Master of Science in Engineering

**DTU Compute**
Department of Applied Mathematics and Computer Science

# SensibleJournal 2014: A Mobile Life-logging System for Personal Mobility

Georgios Chatzigeorgakidis (s121078)

Kongens Lyngby 2014

**DTU Compute**
**Department of Applied Mathematics and Computer Science**
**Technical University of Denmark**

Matematiktorvet
Building 303B
2800 Kongens Lyngby, Denmark
Phone +45 4525 3031
compute@compute.dtu.dk
www.compute.dtu.dk

# Abstract (English)

This text presents a mobile *life-logging* application called SensibleJournal 2014, which was deployed as an upgrade to the 2013 version. It uses server-side calculated stop-location information derived by the user's SensibleDTU GPS sensor gathered data. This information is elaborated in order to render a card-based interface, which provides map-based visualizations regarding a person's personal mobility. SensibleJournal was installed or upgraded to version 2014 by N = 417 experiment participants. An analysis, based on usage information collected from each user's phone was performed, providing several insights about the way people prefer to *self-reflect* on their personal movement data.

# Abstract (Danish)

Denne afhandling præsenterer en *life-logging* mobil-app kaldet SensibleJournal 2014, der konstrueres som en opgradering af 2013-versionen. App'en benytter server-side-kalkuleret stop-location-information, som udledes fra data indsamlet via brugernes SensibleDTU GPS-sensor. Denne information viderebehandles med det formål at udvikle en interface med en kort-struktur, som producerer landkort-baserede visualiseringer af brugerens personlige mobilitet. I eksperimentet opgraders eller installeres SensibleJournal til version 2014 hos N=417 deltagere. Der foretages en analyse baseret på bruger-information indsamlet fra hver deltagers mobiltelefon. Analysen leverer adskillige indsigter i, hvordan brugere *reflekterer* over og spejler sig selv i deres personlige bevægelses-data.

# Preface

This thesis was prepared at the department of Applied Mathematics and Computer Science at the Technical University of Denmark in fulfilment of the requirements for acquiring a M.Sc. degree in Computer Science and Engineering.

This report presents SensibleJournal 2014, a mobile life-logging application for visualizing personal mobility information.

Kongens Lyngby, July 24, 2014

[illegible]

Georgios Chatzigeorgakidis (s121078)

# Acknowledgements

I would like to thank my supervisors, Sune Lehmann, Jakob Eg Larsen and Andrea Cuttone, for their guidance, feedback, technical and theoretical support during the elaboration of this work.

Special thanks goes to my family, for their priceless material and immaterial support.

# Contents

CHAPTER 1

# Introduction

*Moore's law*[1] points out that the number of transistors in integrated circuits doubles approximately every two years. The virtues of this growth, along with the cheaper production values, are rather apparent nowadays, with massive computational power and vast amounts of storage being able to fit in increasingly smaller spaces, allowing the creation of significantly powerful devices of varying size. These devices empower the generation, storage and sharing of more and more information, a fact that characterizes the era that is currently elapsing. This significant increase of information yielding and storage capabilities has driven the attention of many researchers on several scientific areas that were previously considered ungovernable. *Personal informatics systems* is such a research field, which, even older in concept, has attracted the research community and generated an increasing commercial interest over the past few years.

Personal informatics focuses on the ubiquitous or selective collection and process of an individual's personal everyday information, from single or various sources, using either manual gathering methods (e.g. pen and paper) or automatic supervised, or unsupervised methods (e.g. various sensors, such as GPS sensors, cameras, heart rate sensors, bluetooth, WiFi, etc.). The purpose is the interpretation of the collected data, in a way that useful information can be derived. This information is then presented to that individual, using several, easy to comprehend visualization methods, encouraging him/her to *self-reflect* on the results. Self-reflection, one of the most important objectives of personal informatics, is the process of acquiring knowledge about one's nature and purpose, in order to facilitate the answer of specific questions that will lead to better understand him/herself. However, apart from self-reflection, personal informatics can be valuable to numerous other domains, such as health, finance or physical activity.

This work presents *SensibleJournal 2014*, a personal informatics mobile application that uses location data gathered from a smartphone's GPS sensor, in order to render several visualizations placed on a card-based user interface. It is intended to upgrade last year's SensibleJournal 2013 application which was designed, developed and deployed on Google Play Store by the SensibleDTU team [CLL13] [LCL13]. It elaborates the user's stop-locations (places that he/she has visited during the day), in order to derive useful statistical information which are placed on the interface's cards, along with a static mini-map which contains pinpointed locations indicating his/her daily/weekly routes, most visited places, latest movements and other useful

[1] http://en.wikipedia.org/wiki/Moore's_law

information. The stop-locations are calculated server-side, using the raw collected location data. Each card corresponds to a more detailed view, which provides extra information along with an animated visualization, which takes place on an interactive map carrying the same pinpointed locations as the card.

SensibleJournal 2014 was designed and implemented following a prototyping/feedback-design-implementation/feedback-testing scheme, which is used in the development of many desktop/mobile/web applications and websites nowadays. Initially, *paper prototyping* was performed and several design decisions were made, after presenting the prototype to the supervisor and the assistant supervisors of the project. It was followed by *software prototyping*, which led again to receiving useful feedback before going into the design and implementation phase. During that phase, *formative usability testing* was conducted and many aspects of the design were altered according to the feedback. Finally, when the implementation was concluded, the application was handed out to several testers (*summative usability testing*), in order to receive more useful feedback and perform the last alterations according to it. All of the above will be defined and analysed in a later chapter.

As mentioned, the data that the application elaborates, are gathered from the smartphone's GPS sensor. This is carried out by a separate data collection application, called SensibleDTU, which is installed on the phone and executes in the background, gathering data from several sources (GPS sensor, bluetooth, Call Log, etc.). The data are uploaded and securely stored on the SensibleDTU server, where several calculations take place, such as the stop-location determination. Both SensibleDTU data collector and SensibleDTU server are part of the *SensibleDTU* project, whose main purpose is the experimentation on high temporal resolution, large-scale human interaction networks, using data from various sources and will be further defined and described in one of this thesis' chapters. The participants of the SensibleDTU project have received a high-end smartphone for reasons of gathering the personal data required for carrying out the project's experiments, or testing newly developed technologies, such as SensibleJournal.

The final application was deployed on Google Play Store, in order for participants to install and use it. A usage collection subsystem was embodied in the application, responsible for gathering information about the way the participants use it, in an "event-timestamp" manner. The collected usage data were automatically uploaded from each participant's phone on the SensibleDTU server daily, in order to be used for the experimentation phase of this work, which will be presented later. The source code can be found at: `https://github.com/andrea-cuttone/SensibleJournal`.

The rest of this text is structured in the following way:

- Chapter 2 presents a general background for the personal informatics systems and the visualizations' role in them by pointing out and summarizing the most useful definitions about those topics, relative to the SensibleJournal application. It also enumerates several related articles and commercial products. Finally, it outlines the SensibleDTU project and its purpose, along with the components of it that were used in this work.

- Chapter 3 describes the way the application was developed from scratch, from prototyping, to design, to the final usability testing.
- Chapter 4 illustrates the deployment of the application to the Google Play Store and the elaboration of the results gathered by collecting usage information.
- Finally, Chapter 6 presents some observations that were made based on the usage collecting results. It also correlates the SensibleJournal application with the theory and definitions referenced in Chapter 2. Finally, it proposes several future work possibilities and the way they could be achieved.

CHAPTER 2

# Related Work

This chapter goes through several important existing efforts, related to personal informatics and life-logging in an attempt to define and enumerate all the essential aspects about creating such an application. Furthermore, work related to the highly important area of visualizations used in such areas is also presented here. Finally, a reference to the SensibleDTU project and the way is is structured, along with the SensibleDTU stop-location API and the first version of SensibleJournal[1] is also presented in the following sections.

## 2.1 Personal Informatics - Life-logging

### Definition - Theory

Ian Li, Anind Dey and Jodi Forlizzi [LDF10b], defined personal informatics systems as "those that help people collect personally relevant information for the purpose of self-reflection and gaining self-knowledge". *Collection* of and *reflection* on one's personal information are two separate fields that personal informatics systems try to combine: Collection includes the process of gathering the data, in an automated or manual way using various methods, from specific electronic devices (wearables, cameras, GPS devices, heart-monitor devices, smart-phones, etc.), to pen and paper. Reflection is closely connected to the analysis of the collected data and their presentation, using easy to interpret visualizations, or other means, in order to facilitate a person's acquisition of *self-knowledge*. [LDF10b] models the way personal informatics should be organized, in a five-stage schema:

- *Preparation Stage*: Describes the phase where the type of the information to be collected is determined.
- *Collection Stage*: During this stage, a person collects the type of information decided during the previous stage. The collection can be done in an automated, manual, or hybrid way.
- *Integration Stage*: This is the stage where the gathered data is evaluated and processed, in order for a person to be able to reflect on it.
- *Reflection Stage*: During this stage, a person reflects on the processed data from the previous stage. It can be *short-term* (reflecting on the collected data

[1]https://www.sensible.dtu.dk/?page_id=805

immediately after it was collected), or *long-term* (reflecting on the collected data after some time has passed).

- *Action Stage*: This is the stage where the user has acquired self-knowledge through the reflection stage and takes specific actions according to it.

Each one of the stages introduces several *barriers*. For example, preparation stage introduces barriers on the determination of tools to be used for data collecting, or the decision of types of information to be retrieved. Collection stage's barriers are due to difficulties that occur while collecting the information, such as problems with the chosen tools, or lack of time for proper collection. Integration's barriers are of significant importance, as they prevent the user from transitioning to the reflection phase, for example, due to poor interpretation and analysis of the collected data, leading to confusion. Apparently, this could lead to barriers on the reflection stage, such as difficulty to reflect on the data, caused by poor visualization methods. Visualizations are a crucial part of personal informatics systems and will be examined in the next section. Finally, an important obstacle during the action stage is the difficulty in determining the actions to be taken. Systems that have the ability to automatically suggest actions based on the available integrated information is an interesting area of experimentation. The research concluded in four properties of the five-stage schema:

- *Barriers cascade*: They can influence the next stages (e.g. from integration to reflection stage, as described above).
- *Stages are iterative*: One could transit from a specific stage to a previous one, in case the result was not satisfying.
- *User driven vs System driven*: Stages can be either handled by the user, or automatically by the system.
- *Facets*: Personal informatics systems can either produce results about a specific domain in a person's life (uni-faceted) or more (multi-faceted).

Facets property reveals the opportunity for discussion about the types of data that are collected during the collection stage of a personal informatics system. As mentioned above, there are multiple sources from where information can be collected [LDF11]. This generates the definition of *multi-layer* data and poses a significant challenge, which is the automation of the collection/integration/reflection stage using data from multiple sources. Ian Li, Anind Dey and Jodi Forlizzi extended the definition of personal informatics systems [LDF11] "from a class of tools to an activity where people collect and reflect on personal data to gain a better understanding of their own behaviour", giving significance to the difficulties that rise during such an activity and the tendency towards the direction of using specific technologies for facilitating the collection, integration, reflection and action stages. They connect the use of personal informatics systems to behaviour-changing procedures, according to the *Transtheoretical Model of Behavior Change* (TTM) [PV97]. In [LDF11], after conducting a survey, they concluded that people ask six kinds of questions about their personal information:

- *Status*: They desire to be informed about their current status in a specific domain, according to the collected data.
- *History*: They usually want to go through the collected data, in order to remember, find trends or patterns.
- *Goals*: Deriving from the above questions: what goals could one set, according to his/her collected data?
- *Discrepancies*: Again, deriving from the above: What is ones current status, in contrast with the goals he/she has set?
- *Context*: How the context of other concurrent events affect one's present status?
- *Factors*: How the context of older events affects one's behaviour "over a long period of time"?

Abigail Sellen and Steve Whittaker introduce another aspect of personal informatics systems in [SW10], called *life-logging*. They mention: "we distinguish between life-logging and other more deliberate activities involving the capture of personal data (such as digital photography and blogging) that involve the effortful selective capture and display of digital materials for a particular audience. In contrast, life-logging seeks to be effortless and all-encompassing in terms of data capture". In other words, life-logging systems (life-logs) can be thought of as a sub-category of personal informatics systems that aim to completely automatize the previously mentioned collection stage. Two classes of life-logging are introduced:

- *Total Capture*: Life-logging systems that aim to collect data from various source in an as continuous as possible manner. They also are referred to as *ubiquitous* life-logging systems.
- *Situation specific capture*: Life-logging systems that aim to automatically collect complex information containing data during a specific situation.

Abigail Sellen and Steve Whittaker described the potential virtues that life-logging systems (life-logs) can have over the memory of their users, using a model called "*the five Rs*":

- *Recollecting*: Ability to "mentally re-live" a specific situation with a more practical purpose (e.g. remembering where the keys are).
- *Reminiscing*: An extension to recollecting, where one can emotionally re-live specific past experiences.
- *Retrieving*: Ability to retrieve specific digital information, collected by a life-log.

- *Reflecting*: Detecting patterns and specific behaviours using the collected data.
- *Remembering intentions*: Ability to remember a specific action that one was intended to do in the future.

Privacy and security is another major issue of personal informatics and life-logging systems as outlined in [SW10]. Sensible and personal data are required in order for such systems to operate and produce satisfying and meaningful results. [CGG+06] argues about the general understanding of privacy risks that people face every day and the level of its compromise, which is increasing. It is, thus, crucial for a personal informatics system to utilize personal sensible data in a secure and private manner and at the same time, be able to inform its potential users about the threats that their personal data are being exposed to, and the way the system manages it.

## Other Related Work

Personal informatics and life-logging are not new research domains. In 1945, Vannevar Bush described his vision about MEMEX (generally believed to be an abbreviation for MEMory EXtender[2]), a mechanical desk which could archive and index books, articles or any various documents, allowing easy automated retrieval, in order to augment a person's memory. Steve Mann, in 1997 introduced his latest version of WearComp [Man97], a project featured wearable image and audio capturing devices for capturing everyday moments.

Having the ability to reflect on personal information hides many virtues, apart from gaining self-knowledge. Life-logging and personal informatics systems can be used in order to provide help in a variety of domains, such as: finance, health, physical activity, productivity and more [LDF10b]. [WKP+12] indicates the importance of the assistance that such systems can offer to people with memory problems, for example elder people, or Alzheimer patients. They also refer to further opportunities in health monitoring and neuroscience areas. Furthermore, [AEH13] suggests that using location gathered information can aid in food and physical activity behaviours.

Reminiscing and re-living past experiences is a personal informatics research domain that has received a lot of scientific attention over the past few years. [SZC12] suggests that reflecting and reminiscing on common collected data through a personal informatics system may improve a relationship's status. [PTH13] introduced *MyRoR*, a context-aware system which creates text stories of one's experiences (using data coming from various sources - multi-layer) along with several easy to interpret visualizations. [CSS+12] and [AH11], discuss about supporting memory and reminiscing through software personal informatics systems that support the integration stage (*Pensieve*, *Digital Parrot*). [CSS+12] also annotates the importance of the increasing capabilities of mobile devices in data collecting. For the same reasons, [DPTC+12] and [CNBM+12] use Microsoft's *SenseCam* on several experiment participants, in order to automatically group their collected images into events. [KK10] describes

[2]http://en.wikipedia.org/wiki/Memex

*Mneme*, a personal informatics tool which uses data gathered from various sources in order to aid people with organizing their *Sisyphean* tasks, which can be described as repeated everyday routine actions of various importance. Finally, [PHS13] introduces *Wandering mind*, a timeline-enabled, story-based graphical tool built around a theoretical approach on self-reflection, which also provides with the possibility of sharing created stories via several social networks.

The general availability of portable GPS devices in conjunction with the vast increment of GPS enabled smartphones' usage during the past few years, has facilitated the research and development of many location-based personal informatics and life-logging systems. [KSWK10] uses a GPS sensor together with Microsoft's SenseCam in order to create map visualizations of the participants' movements. The latter has many similarities with SensibleJournal, the life-logging system that was developed for this thesis' purposes. [AF11] suggests the usage of GPS collected data for assisting the development of geographically enabled *Personal Information Management* (PIM) systems. [RLJ+11] uses a GPS sensor along with digital a camera mounted on a car to gather data. Using image processing methods, the system recognizes and categorizes everyday driving events. Digital image processing is also used by [TMA+08], which attempts to model the context of multi-layer life-logging data using a *5 Ws* model (who, when, where, what and why). [YI09] detects events using recorded audio data, in conjunction with GPS gathered data, while [TSMN12] introduces *MashMap*, which again, similarly to SensibleJournal, uses data collected from a GPS device, in order to create map visualizations. [AFH12] uses location information along with other types of data gathered using a smartphone, in order to model a person's everyday behaviour (e.g. what he/she is doing, when and with whom). Finally, [TM13] uses *machine learning* methods in order to estimate one's outdoor activities (e.g. what he/she was possibly doing in a specific location).

*Ontology* based personal informatics systems such as [SHKK10] and [LGHL10] are of particular interest, which attempt to model the gathered data through an ontology, created using specific languages, such as *RDF*[3], *RDFS*[4] and *OWL*[5]. This will allow the calculation of useful contextual information estimated by processing (*reasoning*) the ontology data and their *meta-data*, which will create various semantics and correlations between the entities that take part in the process. Ultimately, the processed data will be available through ontology querying languages, such as SPARQL[6]. [KLK+12] and [SKS+13] attempt to predict future behaviour (e.g. estimated arrival time) by recognizing patterns from a collection of multilayer gathered information.

The massive turning of the scientific community towards the personal informatics and life-logging systems has resulted in the development of many commercial products, which gather large amounts of disparate multilayer data from various sources, in

[3]http://www.w3.org/RDF/
[4]http://www.w3.org/TR/rdf-schema/
[5]http://www.w3.org/2001/sw/wiki/OWL
[6]http://www.w3.org/TR/rdf-sparql-query/

order to cover a wide variety of purposes. *Mint*[7] is a (user-driven in collection stage) personal informatics system, operational though a smartphone or through the web, for financial purposes. It automatically organizes spendings in categories and provides an easy-to-use framework for goal choosing. *Nokia Lifeblog*[8] was one of he first smartphone-based life-logs. Launched in 2004, it creates a multimedia diary using photos, videos and SMS messages gathered from the phone. *Friday*[9] and *Narrato*[10] are autobiographical life-logs for Android and iPhone, which create stories based on several data, also gathered from the device. *Saga*[11] and *Moves*[12] are location based, smartphone life-logs. They detect the locations that the user visits (*stop-locations*) and present several stats and a feed about them. They can connect with many other applications, such as *Move-O-Scope*[13], a web based application that presents the movements of a user on a map, using modern visualizations. *The Transit App*[14] is a personal informatics smartphone application which uses location data in order to aid its users with their transportation in several supported cities. *Expereal*[15] is an interesting work in progress aiming to capture the daily feelings of a person and present useful statistics using *DataViz*[16] visualizations. *Mem:o*[17] and *Reporter*[18] are also two smartphone-based storytelling life-logs.

Wearable electronic devices are becoming more popular nowadays and can be used in many ways, in conjunction with health-based life-logging systems. They are connected with smartphone devices usually via bluetooth and are usually equipped with heart monitors in order to facilitate the detection of everyday activities. *Nike+*[19] provides several wearable devices, as well as a compatible software for tracking daily exercise. *Jawbone's Bracelet*[20], also accompanied by a compatible mobile software, aims to provide help related to sleep, food or other every day activities. Furthermore, *Quantid*[21] and *Human*[22] are health-based life-logs for phones (both user-driven and system-driven using wearables during the collection stage), which gather data about daily exercise and aid their users in setting and pursuing desired goals. Finally, of particular interest is [JPL+13], which introduces the use of an *Electroencephalography* (EEG) neuro-headset connected with a smartphone, which provides real-time feedback according to the brain activity via 3D images. The purpose of the research

[7]https://www.mint.com/
[8]http://www.gizmag.com/go/2729/
[9]https://play.google.com/store/apps/details?id=com.dexetra.friday
[10]https://itunes.apple.com/app/narrato-journal/id668158681?mt=8
[11]http://www.getsaga.com/
[12]http://www.moves-app.com/
[13]https://move-o-scope.halftone.co/
[14]https://play.google.com/store/apps/details?id=com.thetransitapp.droid
[15]http://www.fastcodesign.com/1671535/an-app-that-uses-dataviz-to-digitally-track-your-moods
[16]http://www.dataviz.org/
[17]https://itunes.apple.com/us/app/mem-o/id662546859?mt=8
[18]http://www.reporter-app.com/
[19]https://secure-nikeplus.nike.com/plus/
[20]https://jawbone.com/
[21]http://www.quantid.co/
[22]http://human.co/

is the setting of the foundations for designing interface components, based on the information retrieved from such a neuro-feedback process.

It's worth mentioning that, several of the works depicted in this section were presented during four Computer-Human interaction workshops, that took place in several places around the world, from 2010 to 2013 [LDF10a] [LHD+11] [LMFL12] [LGF+13]. The main focus of those meetings was to define personal informatics systems, introduce their limitations, establish practices according to which such systems should be designed and enumerate and analyse the aspects of life that can be improved, and how this can be achieved.

The following paragraph will present related work in visualization, which is, as derived from the above, a significantly important area for personal informatics and life-logging systems.

## 2.2 Visualization

### Definition - Theory

As discussed in the previous section, reflection is one of the five personal informatics systems stages, suggested in [LDF10b]. It is crucial for such a system to be able to provide a "portal", through which the user's self-reflection on the gathered and integrated (third stage) data will be facilitated. The role of this "portal" is, in every case, played by a *visualization*. [CPL14] discusses that, providing aid in the reflection stage using proper visualizations, can facilitate the understanding of the gathered data and lead to a desired behavioural change via a process of "turning observations and insights into actions". In order for this to be realised, it is significantly important for a visualization to stimulate the ability of a personal informatics system to support the six questions for reflection, presented in [LDF11].Conclusively, visualizations are an integral part of personal informatics systems, on which, special attention should be given.

Many personal informatics systems collect data from various sources (multi-layer data), incommoding the development of proper visualizations for a person to reflect on. As [CPL14] suggests, reflection should be treated "as data analysis on personal information" supported by visualizations using "position, size, shape, color and text", in order to enable the human visual system's ability to comprehend the data and detect patterns, or infer other useful information. In this work, Andrea Cuttone, Michael Kai Petersen and Jakob Eg Larsen define a space consisted of all possible questions (*question space*) accompanied by a space of all the possible visualizations that could answer them (*feedback space*). Based on the most self-reflection related enquiries, they propose four heuristics:

- *Make data interpretable at a glance*: The user must be able to quickly interpret and reflect on the data he/she is presented with. Any components (text, axes, legends, annotations or colors) that could aid the comprehension should be used, though carefully, as too much information could make the visualization too complex.

- *Enable exploration of patterns in time series data*: The user should be able to easily detect and interpret repeating events and patterns, that could aid action taking decisions. [LCL13] introduces an interesting visualization that allows for easy detection of patterns. It presents temporal data on a spiral, dividing time units in arcs and events in colors, enabling the detection of periodic events.

- *Enable discovery of trends in multiple data streams*: Allow the user to explore possible relations between data originating from different sources.

- *Turn key metrics into affordances for action*: Use interactiveness in visualizations in order to enable the exploration of the data presented and maybe reach to new decisions/actions, or discover new important metrics for the user to reflect on.

### Other Related Work

The significance of visualizations can be mirrored in the fact that most personal informatics scientific works that present such systems, employ and give a large amount of emphasis in the visualizations of the collected and integrated data. For instance, *Spark* [FFD12] uses abstract art for visualizing step counts, gathered using the *Fitbit*[23] warable device. [LML+06] introduces *Fish n' Steps*, an interactive computer game that uses a graphical fish in a bowl visualization for everyday data, gathered by pedometer devices. *UbiFit Garden* [CMT+08], is another life-logging system, which uses on-body sensing and real-time statistical modelling in order to visualize daily physical activity using a smartphone. Finally, [FDK+09] presents *UbiGreen*, a mobile application which updates the wallpaper of the phone using graphical images that represent green transportation behaviour, using data gathered from the phone's sensors.

Many of the, already mentioned in the previous section, research studies also use several visualization methods for aiding the participants' self-reflection on their data. *Lifestyle Stories* [PTH13] is such an example, which uses data gathered from various sources to create daily stories, which can be visualized through text and images. Another similar work is *Wandering mind*, a tool introduced in [PHS13], which aims to support reminiscing and self-reflection in general, through a timeline-enabled interface providing story-based visualizations using images, locations, text, or other types of data. [KSWK10] is another example, featuring *Tracks*, *Snaps* and *SnapTracks*, which visualize movements of the participant by pinpointing locations on a map (Tracks), show SenseCam gathered images in a sequential manner (Snaps), or a combination (SnapTracks). Finally, [RLJ+11] implements a visualization which presents images of the detected events during driving, along with pinpointed map locations and informative text for each event.

[23]http://www.fitbit.com/uk

Concluding, it's worth pointing out that, all previously presented commercial products (*Mint*, *Nokia Lifeblog*, *Friday*, *Narrato*, *Saga*, *Moves*, *Move-O-Scope*, *The Transit App*, *Expereal*, *DataViz*, *Mem:o*, *Reporter*, *Quantid*, *Human*) offer varieties of polished visualizations.

## 2.3 SensibleDTU

*SensibleJournal 2014*, the application implemented in the context of this thesis, was developed under the *SensibleDTU*[24] project, housed in Technical University of Denmark. SensibleDTU is based on *High Resolution Networks* and *Social Fabric* projects, supported by the *Villum Foundation* and *University of Copenhagen* respectively. It is a part of a large scale study, named the *Copenhagen Networks Study*. As suggested in a recently published paper [SSS+14], SensibleDTU's purpose is the experimentation on high-resolution, large-scale human interaction networks which are formed by gathering data from various sources, in order to study human individual and group behaviour on a large chronological scale. This is achieved through the SensibleDTU system (two versions deployed up to date - 2012 and 2013) which aims to collect multi-layer data (via questionnaires, face-to-face interactions, telecommunication, social networks, location and background information, always giving proper emphasis to privacy), from a "densely connected" population, allowing the conduction of several experiments, such as the analysis of the network dynamics that is formed between the participants. What SensibleDTU aims in long-term, is the analysis of data of entire generations, since ubiquitous collecting of information is becoming more and more feasible.

### Data Collector

As mentioned, multiple-source data are collected in the context of SensibleDTU project. In particular:

- *Questionnaires*, handed out to participants
- *Facebook Data*, optionally, through the participants' accounts
- *Sensor Data*, via smartphones handed-out to 1000 participants
- *Qualitative Data*, via an anthropological field study
- *WiFi Data*, through the campus' WiFi system logs

Since SensibleJournal 2014 is a mobility logging application, it was built upon information derived from the sensor data. The collection takes place via an Android application deployed on the Google Play Store, called *Sensible DTU 2013*[25], which

[24]https://www.sensible.dtu.dk/

[25]https://play.google.com/store/apps/details?id=dk.dtu.imm.datacollector2013

is based on the *FunF Open Sensing*[26] framework. Similarly to the SensibleDTU system, two versions of the Sensible DTU data collecting application were deployed: one in 2012 which was updated with the new version in 2013. The latter, which is the one that is used currently, authenticates the users through an OAuth2 authorization system, which is initiated upon the first launch of the application. Upon login, the data collector notifies the user about his/her personal data that will be collected. It gathers the data from the phone, caches them locally and automatically attempts to upload them on the SensibleDTU server every two hours, only if there is an active WiFi connection. The uploading can also be performed manually through the application. The collector creates a unique token identifier, embedded in the data files, which after being uploaded to the server, are decrypted, compressed and finally stored. The data are collected using fixed intervals and are gathered from the below sources:

- Location data, via GPS
- Bluetooth data
- Call logs
- SMS
- WiFi data

### Infrastructure

The SensibleDTU backend (2013 deployment), housed at the Technical University of Denmark, is an *OpenPDS*[27] based system. The way it is developed, allows for easy extensibility and the ability to support different services, as depicted in Figure 2.1 (source: [SSS+14]). The architecture of the system, shown in the figure, is consisted of the following layers:

- *Platform*: This layer contains the common components of SensibleDTU's services.
- *Services*: Each supported service's distinct components are grouped in this layer.
- *Applications*: Contains the applications associated with each one of the services.

[26]http://www.funf.org/
[27]http://openpds.media.mit.edu/

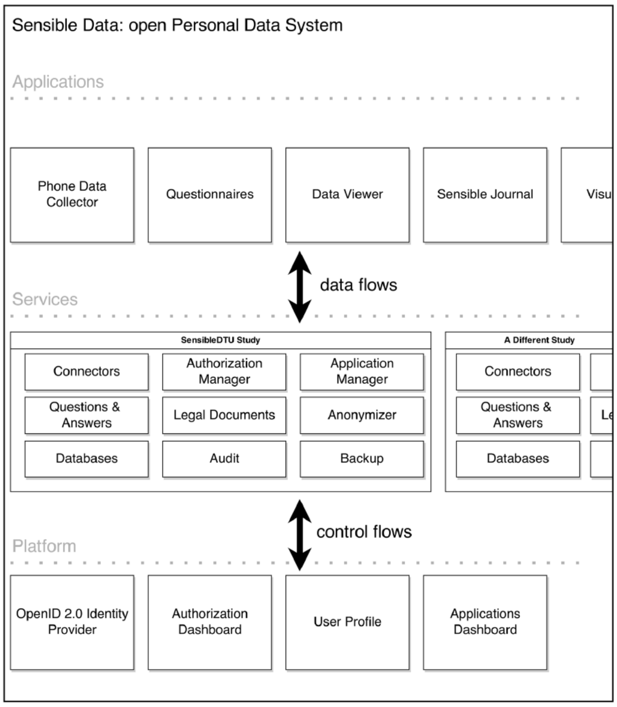


Figure 2.1: SensibleDTU's (2013 deployment) architecture

### SensibleJournal 2013

As already discussed, the Android application developed during this thesis, called SensibleJournal will replace the 2013 version which was already deployed in Google Play Store, developed by Andrea Cuttone [CLL13] and supervised by Sune Lehmann and Jakob Eg Larsen. SensibleJournal 2013 adopted a tabular view and used GPS and bluetooth data collected by the SensibleDTU data collector application, in order to create movement and social related visualizations. Figure 2.2[28] illustrates the movements tab during a commute animation. SensibleJournal retrieves the necessary data from the SensibleDTU server in JSON format using an HTTP based web-API, caches them locally and performs the necessary calculations, in order to create the visualizations.

### Stop-location API

The stop-location API [CLL14] (based on [ZZXM09]), is an HTTP based web-API, recently integrated into SensibleDTU, whose purpose is the analysis of the GPS data,

[28]Image source: https://www.sensible.dtu.dk/?page_id=805

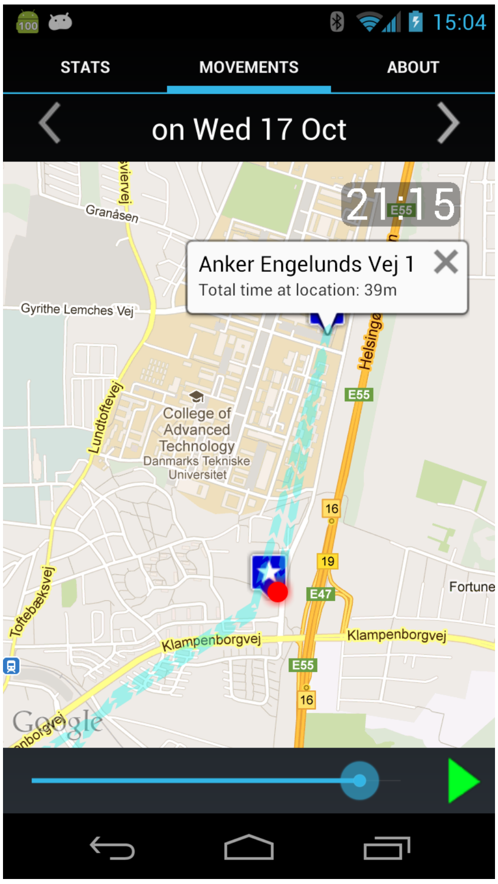


Figure 2.2: Movements tab of the 2013 version of SensibleJournal

collected from the participants' smartphones and derivation of the *stop-locations*, which are places that the user visits for amount of time larger than a specific interval. It operates through an HTTP request and returns the stop-locations along with other meta-data in JSON format. After the gathered data are uploaded to the SensibleDTU server, the stop-locations are extracted as groups of consecutive locations within a predetermined distance and then are clustered using the *DBSCAN*[29] clustering algorithm. This operation takes place for all the newly gathered data once per day, approximately at 6:00am. The calculated stop-locations are stored on the server in a latitude-longitude format, along with several useful meta-data, such as time of arrival/departure and a unique label for each location.

It is worth mentioning that, the web-API that was used in SensibleJournal 2013

[29]http://en.wikipedia.org/wiki/DBSCAN

would return all the collected GPS data, which could result in server overloading occasions as well as intense computations on the phone. In contrast, SensibleJournal 2014 uses the stop-location API, the usage of which will be further discussed in the following chapter, along with the way the application was designed, implemented and tested.

CHAPTER 3

# Design & Implementation

This chapter describes the design and implementation of SensibleJournal 2014, from the early prototyping stages (Section 3.1), to the final coding and testing of each of its components (Section 3.2). In addition, any algorithms and visualization techniques that were used in order for the final result and user experience to be achieved are also described in Section 3.2.

## 3.1 Prototyping

*Prototyping* is a procedure of continuously increasing significance, that takes place before any implementation action. Over the past few years, it has become popular in the website/mobile/desktop application development community, as it provides many ways of representation and simulation of a possible future release of the application. Demonstrating a prototype to testers, helps avoiding a large amount of editing or redesigning that might occur during the implementation phase. This procedure is part of the *formative evaluation* of the application, as "it takes place during the design process" [Lar13].

As far as the mobile application development is concerned, the most widely used types of prototyping are *paper prototyping* and *software prototyping*. Paper prototyping includes the (usually) very first group of actions that can be taken during the design phase of an application, or an element of an application. It involves drawing the *user interface* (UI), as well as the transitions between different views/screens/-menus of the application using pen and paper, covering as many usage scenarios as possible. The resulting prototype can the be demonstrated to a number of testers in order to receive useful feedback, before moving to the next design phase. Each element of the prototype should be explicitly described using text, or verbally while demonstrating. The designer presents the prototype's drawings to the testers and a verbal interaction takes place, simulating the final application's functionality. This procedure is known as *Low-Fidelity Usability Testing* [OHRM09]. It is usually a recurring process, as the prototype can be sketched, presented and critiqued until a valid, satisfactory outcome occurs [Tra12]. The virtues of paper prototyping become more apparent considering the freedom and simplicity that a simple pen and paper design can offer. Brian Still and John Morris took it a step further by allowing the testers to draw what they would expect to see while navigating through a web-site prototype [SM10].

Paper prototyping in mobile applications is usually followed by software prototyping, where several authoring tools can be used, in order to create an interactive wireframe of the application. The software prototype provides primitive transitions between the menus and views/screens of the application and in general, simulates the way the application will operate. As in paper prototyping, the prototype is demonstrated to a number of testers in order to obtain useful feedback and proactively fix or alter any elements that require so, before moving to the implementation phase. This procedure is known as *High-Fidelity Usability Testing*. There is a handful of applications that can create high-fidelity software prototypes. *Proto.io*[1], *fluid*[2] and *prott*[3] are mostly used for mobile application prototyping. *Justinmind*[4] can be used both for mobile and web based prototypes, while *Indigo Studio*[5], *Mockingbird*[6], *ForeUI*[7] and *Solidify*[8] are more generic prototypers, offering the possibility to create desktop application or website prototypes.

Both paper and software prototyping were used while designing SensibleJournal 2014 and will be further analysed in the following two sub-sections.

### Paper Prototyping

As mentioned, the application will make use of the users' stop-location data, which are calculated server-side using their smart-phones' GPS-sensor-gathered movement data. Each user's stop-location data will be fetched from the SensibleDTU server, and used locally by the application. It was decided that the application should support card-based visualizations, placed on a card-feed enabled user interface, which is an increasingly trending method for creating user-friendly web, or mobile applications[9]. The stop-location data should be used in order to create cards, which will present several spatio-temporal information and statistics about the user's movements, pin-pointed on mini-maps placed on each card. With the above in mind, several paper prototypes were created, describing possible card interface implementations. Figure 3.1 shows an example of a card prototype.

As shown in the figure, the card will present a specific day's user's *route* (called "itinerary" in the prototype), which is constructed by the stop-locations he/she visited during that day, in a sequential manner. Some further statistical information, such as the total distance that he/she travelled, as well as the total number of discrete places he/she visited are also conferred in the prototype. It would be convenient for

[1] http://proto.io/
[2] https://www.fluidui.com/
[3] https://prottapp.com/
[4] http://www.justinmind.com/
[5] http://www.infragistics.com/products/indigo-studio/
[6] https://gomockingbird.com/
[7] http://www.foreui.com/
[8] http://www.solidifyapp.com/
[9] http://insideintercom.io/why-cards-are-the-future-of-the-web/

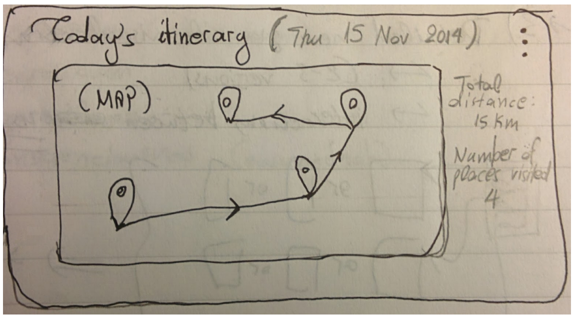

Today's itinerary (Thu 15 Nov 2014)
(MAP)
Total distance: 15 km
Number of places visited 4

Figure 3.1: Paper prototype of a card, showing a specific day's movements

the user to be able to search about itineraries that he/she followed during a certain day in the past. Therefore, it was decided that the application should provide the user with an *archive* view, where he/she would be able to search for past movements. Figure 3.2 shows paper-prototyped possible archive view.

The archive should provide an easy and transparent way for the user to be able to access his/her past movements. The above paper-prototyped elements were gathered along with several others and a low-fidelity usability test took place by presenting them to the supervisor and assistant supervisors. Among them, the most relevant where chosen to be further implemented. The card categories that were chosen were:

- Facts-Cards
  - *My Current Location*: The user's current location's address, along with a map indicating it.
  - *Last Visited Place*: The last place visited by the user (address), along with a map indicating it.
  - *Latest Journey*: The user's latest itinerary indicated on a map, along with further information about it.
- Statistics-Cards
  - *Daily Route*: The user's each day's route indicated on a map, along with further information about it.
  - *Weekly Route*: The user's each week's route indicated on a map, along with further information about it.
  - *Most Visited Places*: The places (stop-locations) mostly visited by the user, along with a map indicating them.

*Facts-cards* would use the latest data available in order to provide informative visualizations about the user's latest movements, while the *statistics-cards* would also use older data in order to provide statistical information derived by the user's movement history (daily, weekly, all-time).

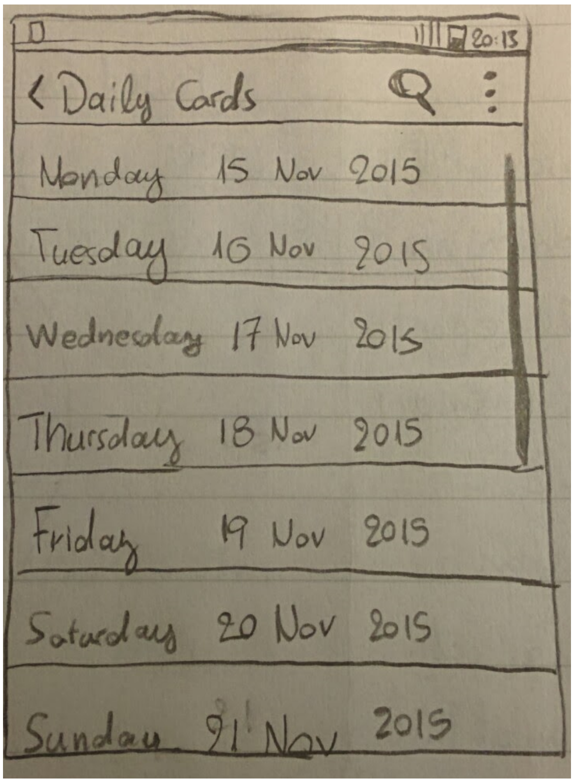
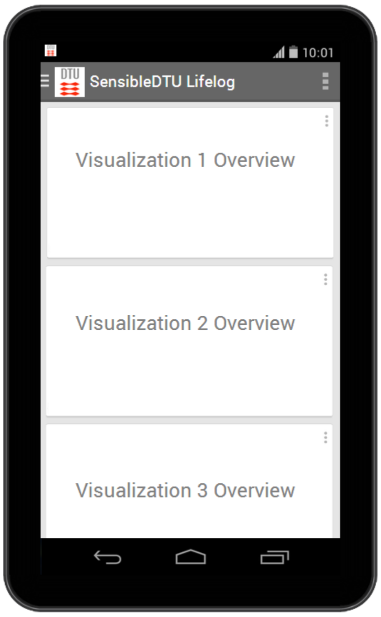


Figure 3.2: Paper prototype of the archive view for daily itineraries

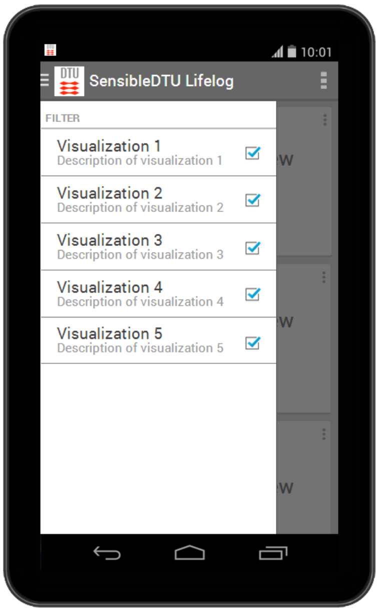
### Software Prototyping

Following the decision among the cards to be implemented and the general interface of the application, a high-fidelity software prototype was created, allowing the tester to interact with its main view, as well as enter some of its menus. In particular, the tester could interact with the list of cards (card-feed) which appears when the application is launched. The tester, also could access the *navigation drawer* of the prototype, which provided with a filter functionality for the application's cards as shown in Figure 3.3(b). The user should be able to choose which cards to be shown in the card-feed of the application's main view (Figure 3.3(a)). This feature was deprecated, for reasons that will be explained later.

At this point it was decided, that each card should be clickable, providing access to a more detailed visualization which would contain additional information. This appeared to be a convenient choice during the development phase, due to the fact that the library that was used for the cards' implementation does not provide with the ability to attach animation-enabled elements on them, except only static images. To



((a)) Main view


10:01
SensibleDTU Lifelog
FILTER
Visualization 1
Description of visualization 1
Visualization 2
Description of visualization 2
Visualization 3
Description of visualization 3
Visualization 4
Description of visualization 4
Visualization 5
Description of visualization 5

((b)) Navigation drawer

Figure 3.3: Software prototype

sum up, clicking on a card would provide each card's corresponding *detailed view* as it was later named (Figure 3.4), which would offer additional interactive, animation-enabled and more descriptive visualizations.

## 3.2 Implementation

As described in the previous section, the initial steps taken for the application's development were paper and software prototyping techniques, which aided in determining the way the overall interface of the application should look like, as well as several fundamental components and transitions that would be necessary. The next step was the implementation of the above decisions in an *Android OS* application. The coding of the application took place in the *Eclipse Integrated Development Environment* (IDE), using the *Android Software Development Kit* (SDK) with the Android *API level 19* (Android 4.4 KitKat). The application was developed on a *LG Nexus 5* device, however the ones that were handed out to the experiment participants were the *LG Nexus 4*, which was released with the Android API level 18 (Android 4.3 JellyBean). There are also other architectural differences between the two, such as the screen density. Consequently, in order to avoid incompatibilities, the application was developed to support both Android API levels 18 and 19, as well as both screen

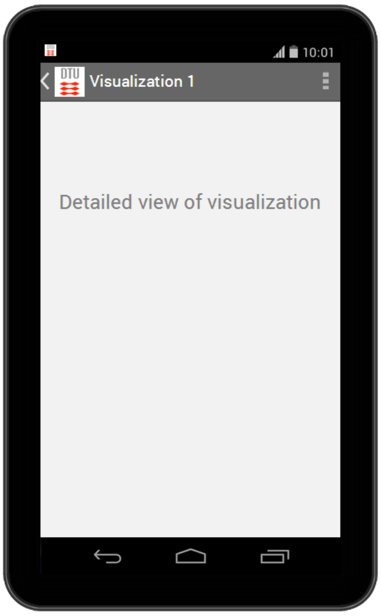


Figure 3.4: Software prototype of a card's corresponding detailed view

densities. This was facilitated by following specific UI design techniques from the *Android design* tutorial[10], which was, in general, followed as closely as possible during the development. This section analyses the implementation of the application and all its elements.

### Server communication

Probably the most fundamental part of the application is the communication with the SensibleDTU server. This is where each user's stop-location data are fetched and processed in order to render the user interface of SensibleJournal 2014. The fetching of the data was implemented through a *Representational State Transfer* (REST) client, which connects to the SensibleDTU server through a *Secure Sockets Layer* (SSL) connection. In order for the server not to be overloaded, the data are stored locally in an *SQLite* database. The way the server communication/data caching operates, is illustrated in Figure 3.5.

The following paragraphs describe the complete procedure.

#### Login

As mentioned, each experiment participant has been handed a Nexus 4 smartphone,

[10]http://developer.android.com/design/index.html

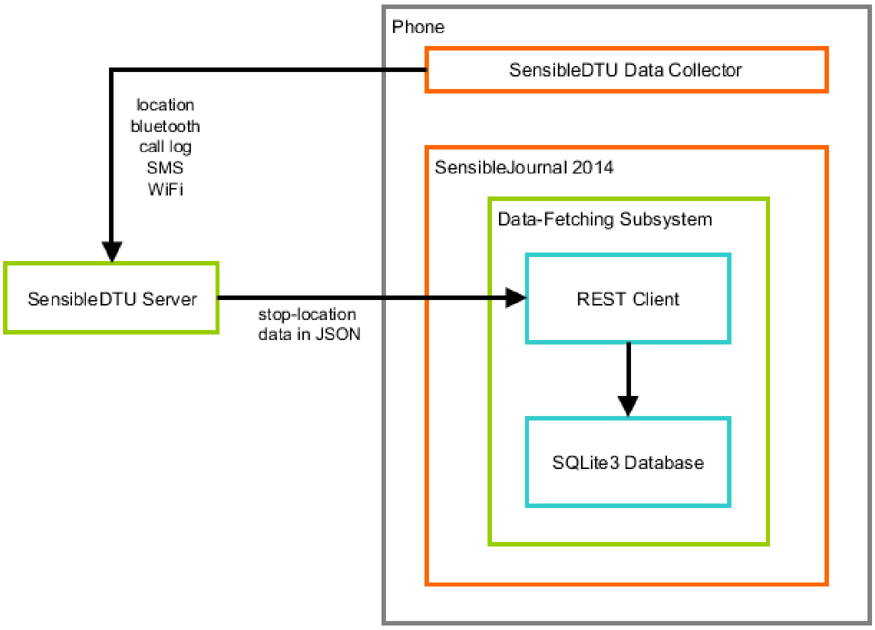


Figure 3.5: The server communication framework

which had the SensibleDTU Data Collector pre-installed. The latter periodically gathers various data from the device (location, bluetooth, call log, SMS and WiFi data, along with the proper timestamps) and uploads them to the server (Figure 3.5). For this to take place, the user must be authenticated via the collector application, using his/her SensibleDTU credentials.

In order to obtain the data that correspond to each user, the same authentication module from the SensibleJournal 2013 and Sensible DTU 2013 data collector was attached to the application. The authentication procedure is launched when the user starts the application for the first time. After providing his/her correct credentials, he/she is prompted with a page which contains information about the data that will be exchanged with the server. For privacy reasons, the user has to accept the data exchange in order for the application to start. After a successful login, the user is granted with a unique 30-digit hexadecimal token, which is stored locally in a file-path accessible only by SensibleJournal 2014. This token is used for obtaining the data from the server, as described in the next paragraph.

**REST Client**

The GPS sensor data that are fetched according to the above procedure are processed server-side, in order to determine the stop-locations. As discussed in the previous chapter, this operation takes place once per day at 6:00am. That being the case, a *data-fetching* subsystem was created, which automatically requests new data at a random time between 10:00 and 13:00 each day (Figure 3.5). This way, overloading the server with multiple requests is avoided, due to the fact that it is rather unlikely for two instances of the application to request new data at the same time. The data-fetching sub-system was implemented using the native Android's alarm manager, which can be set to run a certain part of code in the background, at a specific time. In this case, a repeating alarm responsible for running the data-fetching code is set once during the first launch of the application. Due to the fact that the alarms are cancelled if the device reboots, a system that re-enables them upon booting was developed. If a successful communication between the device and the server has occurred and new data have been downloaded and cached, a flag is raised by the data-fetching sub-system indicating that new data have arrived. Thus, in case the application is brought in the foreground, the interface will be re-rendered using the new data. Fetching from the server also takes place when the application is terminated for some reason (e.g. first launch after install, the device is run out of memory or just manually terminated) and then launched.

The data request takes place using an SSL secure[11] REST client (Figure 3.5) which queries the stop-location API for new data. The queries to the API have the below format:

```
"https://www.sensible.dtu.dk/sensible-dtu/connectors/connector_answer/ +
 v1/stops_question/stops_answer/" + token + "&page=" + page_number"
```

Listing 3.1: Stop-location API request

As shown in the above listing, the stop-location API request requires the user-specific token that was created after the user successfully logged-in. The results are paged and can be fetched using the "page" variable. A start-date and end-date can also (optionally) be provided, if stop-locations within a specific time-interval are required.

In case there is an error during the data-client communication, it is caught by the data-fetching subsystem and a new fetching attempt is scheduled, ten minutes after the failure. This is repeated until there is a successful communication. The ten minutes interval was chosen so that the application won't drain the device's battery quickly. The data-fetching subsystem, in order to complete the communication wakes the device's CPU for as long as required. Then the CPU sleeps again. Through testing procedures, the battery drain appeared to be minimal in case of no internet connection (continuous data fetching attempts).

### User Data

As already discussed, each user's gathered data are processed server-side in order to derive the stop-locations, along with their meta-data. They can be retrieved using a

[11]http://www.makeurownrules.com/secure-rest-web-service-mobile-application-android.html

SensibleDTU API REST request, which contains the unique token for the corresponding user in a paged manner. Each page contains 50 stop-location entries in *JSON* format. An example of an entry is shown in the following listing:

```
{"arrival": 1384012800,
 "lat": 55.6961945,
 "lon": 12.443706500000001,
 "departure": 1384082100,
 "label": 2}
```

Listing 3.2: Stop-location entry

Each stop-location entry is characterized by its *latitude* and *longitude*, as gathered by the device's GPS sensor. In addition, an arrival and departure timestamp are also attached to it, indicating the exact time that the user arrived or left from the specific location. The time is measured using the *Unix epoch* unit, which is the amount of seconds that have passed since the 1st of January 1970 GMT (*Greenwich Mean Time*) at midnight (00:00). For example, Unix epoch time 1401352014 is: Thursday, 29th of May 2014 at 08:26 GMT. The dates are automatically converted to the local timezone, according to the phone's location. For Copenhagen it is GMT+1 standard and GMT+2 *daylight saving time*. Finally, each stop-location entry is assigned a unique integer label.

**Storage**
According to the preceding paragraphs, the stop-location data are fetched once per day in the background, using the data-fetching subsystem and stored locally in an SQLite3 database (Figure 3.5). Thus, when the application is launched (or brought in the foreground), the data are read straight from the local cache and the interface is rendered. Each time the data-fetching subsystem downloads new data, it parses the fetched, JSON-formatted stop-location collection, extracts the metadata for each stop-location (arrival, latitude, longitude, departure, label) and stores them in the corresponding columns in the database. Also, the day of each of the cache database entries is determined, along with an increasing day counter and stored in the proper columns ("day" and "day_number"). This aims to facilitate the fetching of only the entries that belong to specific days, or specific incremental day numbers. The database is handled by the Android's SQLite database helper system, which stores it in the device, in a location only accessible by SensibleJournal 2014, thus avoiding any privacy hazards.

## Metrics

In order to calculate all the required statistics and render the interface, the application has to determine several metric values using the stop-locations along with their metadata, such as the distance between two locations and the commute speed from one stop-location to the next. The latter is also used to determine the means of transport between two consecutive stop-locations.

**Distance**

The distance is calculated using the *haversine* formula, which produces the *great circle* distance between two points. The latter can be thought of as the shortest possible distance between two points on the surface of a sphere. Considering, in this case, that the sphere is the earth, the formula will calculate the shortest distance between two stop-locations, ignoring possible slopes (mountains, hills) in between[12]. The haversine distance is given by:

$$d = 2 \cdot r \cdot \arcsin(\sqrt{haversin(\phi_2 - \phi_1) + \cos(\phi_1) \cdot \cos(\phi_2) \cdot haversin(\lambda_2 - \lambda 1)})$$

where $\phi$ is the latitude and $\lambda$ is the longitude of each stop-location, r is the earth's radius and *haversin* is given by:

$$haversin(\theta) = \sin_2(\theta/2) = (1 - \cos(\theta))/2$$

Consequently, the haversine distance can be written as:

$$d = 2 \cdot r \cdot \arcsin(\sqrt{\sin^2((\phi_2 - \phi_1)/2) + \cos(\phi_1) \cdot \cos(\phi_2) \cdot \sin^2((\lambda_2 - \lambda_1)/2)})$$

**Speed - Means of Transport**

Using the calculated distance between two consecutive stop-locations A and B and the time difference between the departure from stop-location A and the arrival to stop-location B, SensibleJournal 2014 is able to calculate the user's average travel speed between the two points. The calculated speed is then used to determine the means of transport that the user utilized for his commute. The application recognizes four different types of transportation:

- Walking
- Biking
- Car/Bus/Train
- Plane

The human walking speed is between 5 km/h and 9 km/h[13]. Considering that the user might stop from point to point, and after some testing with real data, the upper walking speed threshold was set to 5 km/h. The biking average speed is 15.5 km/h[14]. For the same reasons, the biking speed threshold was set to 13 km/h. Any movement with speed above 13 km/h and up to 200 km/h is considered to be via a car/bus/train, while above 200 km/h is considered to be via air-plane. To sum up:

[12]http://en.wikipedia.org/wiki/Haversine_formula
[13]https://en.wikipedia.org/wiki/Preferred_walking_speed
[14]http://en.wikipedia.org/wiki/Bicycle_performance#Typical_speeds

- Walking: 0 km/h - 5km/h
- Biking: 5 km/h - 13km/h
- Car/Bus/Train: 13 km/h - 200km/h
- Air-plane: 200 km/h -

Using the above values, the application guessed the actual travel means of transport with a satisfyingly high accuracy. Of course, there are many occasions when the vehicle determination is wrong, such as when there is heavy traffic. The cause of this is the fact that the application operates only on stop-location information, which is not adequate for precise means of transport estimation. The distances are calculated using the haversine distance disregarding the actual route, which does not allow accurate speed determination. Hence, this method can be thought of as a reasonable approximation for detecting the way people commute.

### Cards

Upon launching, the application initiates its main *activity* and connects to the *Google Play Services* in order to retrieve the user's current location. Following this procedure, the six cards mentioned in Subsection 3.2 are generated in a separate thread, in order for the application's *Graphical User Interface* (GUI), which is handled by the main thread, not to be heavily loaded (resulting in freezing the application) until the calculations are complete. The cards are implemented using Gabriele Mariotti's *cardlib* library[15]. Each card fetches the data it requires by performing the proper query on the cache database and then is added to the card list, which consists the application's main interface, shown in Figure 3.6.

As illustrated in the figure, the application uses a colouring scheme that matches the colors of DTU. Also, its icon remained the same as in the previous version (SensibleJournal 2013). As already mentioned, the cards that the cardlib library creates are static, meaning that they don't support animations. Therefore, the mini-map that each card contains, is a static-map thumbnail created using the *Google Static Maps V2* API (See Appendix A). The latter is able to create map images including colourful markers, lines and other graphical elements, using an HTTP request.

It should also be noted that all the cards contain an "Awesome!" button which can be tapped in order to provide useful feedback for determining which cards were considered most important by the participants. The following paragraphs will describe each one of the application's cards and the information they provide, in the same sequence as they appear in the application.

[15]https://github.com/gabrielemariotti/cardslib

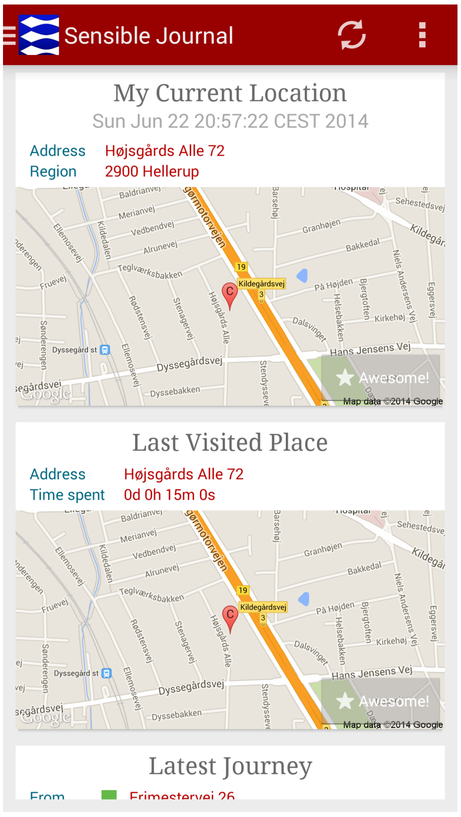
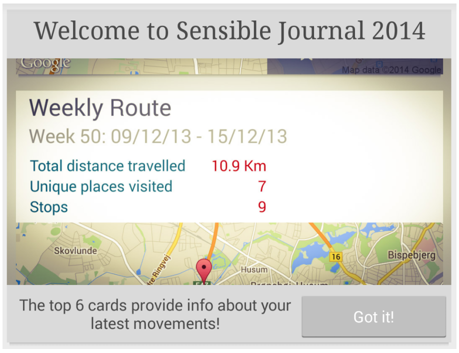


Figure 3.6: The SensibleJournal 2014

**Tutorial Cards**
With the purpose of making the application more user-friendly, several tutorial cards were created that are rendered the first time the user launches SensibleJournal 2014. The tutorial cards contain screenshots of several of the application's elements, along with minimal helpful tips concerning its usage. They are equipped with a "Got it!" button, which, upon tapping, permanently dismisses each card. Additionally, they can also be dismissed by being swiped-away. Finally, they are coloured in a different manner than the normal cards, for reasons of distinguishability. An example of a tutorial card is shown in Figure 3.7.

**My Current Location**
As discussed, the application creates six different types of cards, which are separated in two categories, the *facts-cards* and the *statistics-cards*. Each time the application is launched, the facts cards are rendered first, as they concern more recent events. The first facts-card, which is the top card on the list, is the *"My Current Location"* card,


Welcome to Sensible Journal 2014

Weekly Route

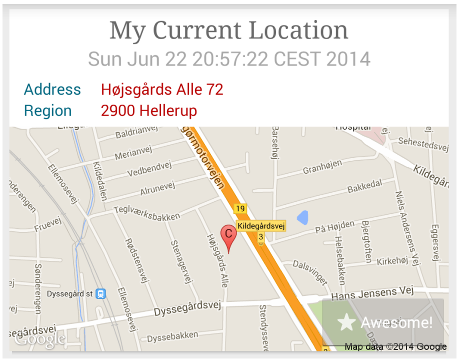
Week 50: 09/12/13 - 15/12/13

| | |
|---|---|
| Total distance travelled | 10.9 Km |
| Unique places visited | 7 |
| Stops | 9 |

The top 6 cards provide info about your latest movements!

Figure 3.7: A tutorial card

which uses the previously fetched current location of the user in order to pinpoint it on a map. It obtains the location's address via the *Google Maps Geolocation* API and reports it in a text area above the static map. An example of the card is shown in Figure 3.8.


My Current Location

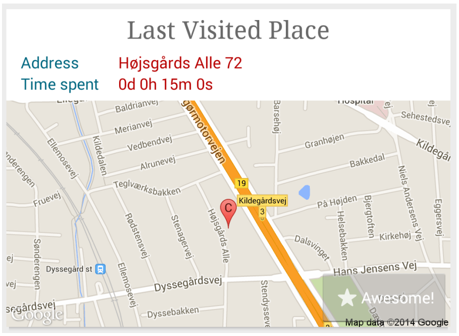
Sun Jun 22 20:57:22 CEST 2014

| | |
|---|---|
| Address | Højsgårds Alle 72 |
| Region | 2900 Hellerup |

Figure 3.8: The "My Current Location" card

## Last Visited Place

The "Last Visited Place" card operates in a similar manner as the "My Current Location" card. Its purpose is to indicate to the user the latest stop-location that he/she visited. The card reads the data for the last visited stop-location from the local cache; it is the latest entry in the database. The data are fetched, analysed and the address of the latest visited location is calculated (using the Google Maps Geolocation API), along with the duration of the visitation. The latter are reported in a text area above the static map image pinpointing the location. Figure 3.9 indicates an example of a "Last Visited Place" card.



Last Visited Place

Address Højsgårds Alle 72

Time spent 0d 0h 15m 0s

Figure 3.9: The “Last Visited Place” card

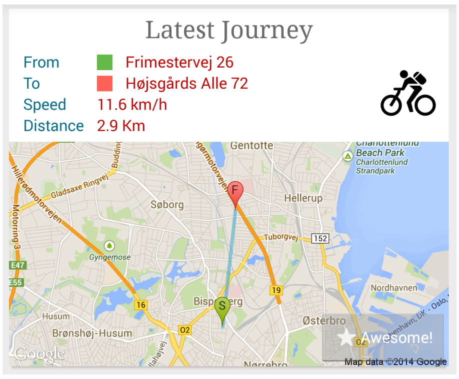
**Latest Journey**

The last facts-card is called “Latest Journey”. It is responsible for presenting information about the latest commute that the user has accomplished. This card also requests data from the local cache. More specifically, the two first stop-location entries are fetched, as they are the ones that represent the latest commute. It should be noted that the data are stored and fetched in a descending manner, according to the arrival to each stop-location. Therefore, if stop-locations A and then B are fetched from the database, the commute will be from location B to location A, which will be the last place that the user visited (see previous card). As in the previous ones, the card fetches the static map indicating the user’s route using the Google Static Maps API. The starting point is pinpointed using a green marker, while the ending point using a red. “Latest Journey” also contains the addresses of the two locations, the average speed and the haversine distance between the two locations. Finally, the means of transport used for the journey is also estimated and rendered using an icon at the top-right part of the card. Figure 3.10 shows an example of a “Latest Journey” card.

**Daily Route**

The “Daily Route” is the card list’s first statistics-card. Its purpose is to inform the user about a whole day’s journey. That day is the latest day that appears in the application’s cache. It contains a static-map, where it pinpoints, using markers, all the stop-locations that the user visited during the day. As in the “Latest Journey” card, it uses a green marker for the initial stop-location and a red marker for the final destination. The in-between stop-locations are pinpointed using blue markers. The card also contains information about the total distance that the user covered, the number of unique places visited and the number of stops between the starting and finishing location. Finally, the day that the card concerns is reported below its title, as depicted in Figure 3.11.

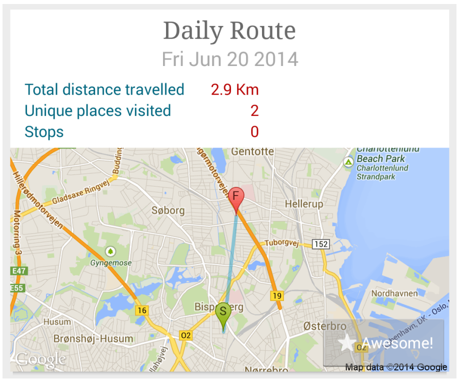
One could ask at this point: “which stop-locations correspond to a certain day’s route?” A useful example, clarifying this question is the following: The user could



Latest Journey

| | | |
|---|---|---|
| From | 🟩 | Frimestervej 26 |
| To | 🟥 | Højsgårds Alle 72 |
| Speed | 11.6 km/h | |
| Distance | 2.9 Km | |

Figure 3.10: The "Latest Journey" card



Daily Route
Fri Jun 20 2014

| | |
|---|---|
| Total distance travelled | 2.9 Km |
| Unique places visited | 2 |
| Stops | 0 |

Figure 3.11: The "Daily Route" card

arrive at a stop-location at some time during Thursday, stay there for a while, depart and arrive at another stop-location at the same day; spend there the whole Friday and then depart again at Saturday for another stop-location. It was decided, for such cases, that the first stop-location should correspond to Thursday's, as well as Saturday's itineraries, as it contains information for both days. However, there was only one entry for that specific location. For this obstacle to be overcame, the application detects (prior storing to the cache) stop-locations that may have different arrival and departure days and uses them as ending-points for the first and starting-points for the second day. This is achieved by storing those stop-locations twice in the cache: one for each of the two days (distinguished by the "day" field in the cache database). Such a case is shown in Figure 3.12.

**Weekly Route**
The second statistics-card is called "Weekly Route". Its purpose is to inform the user

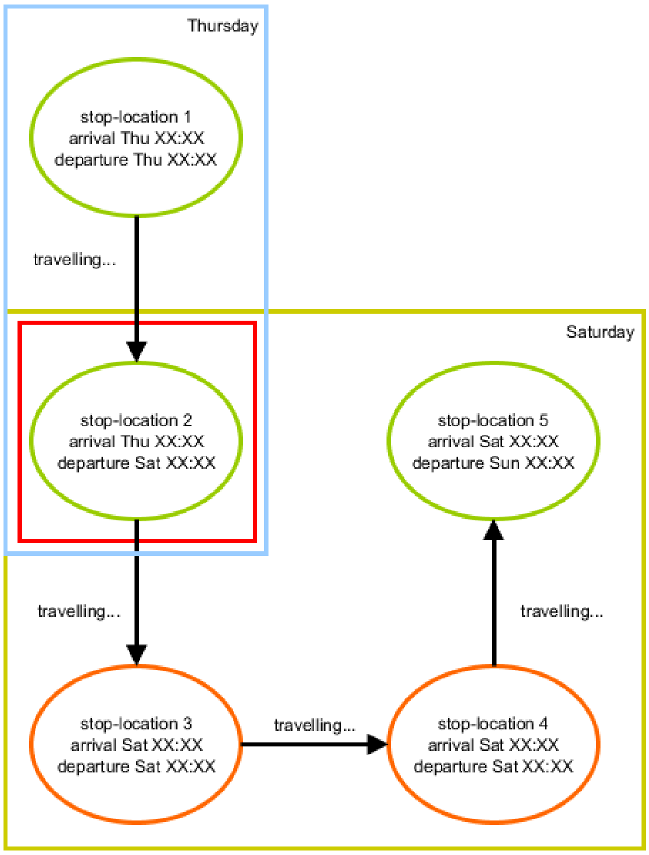


Figure 3.12: Entry that corresponds bot to Thursday's and Saturday's route

about his/her movements during the last week. For the card to be created, the first week's stop-location entries are fetched from the cache and are handled as a complete itinerary. As shown in Figure 3.13, the card is very similar to the "Daily Route" card apart from the fact that all the markers that pinpoint the stop-locations on the static-map are red. This was done for reasons of avoiding very large HTTP URLs, due to the fact that the Google Static Maps API supports URLs of size up to 2048 characters[16]. Requesting static-maps with a week's movements information, can create very large URLs, especially in the case that there are a lot of locations to pinpoint, since the coordinates are float numbers with many decimals. For this to be avoided, apart from removing marker color request for every location, the coordinates were truncated, taking into account only the three most significant decimals for each coordinate value.

[16]http://stackoverflow.com/questions/4278053/google-static-maps-url-length-limit

This ensures safely small URLs and adequate precision to the original coordinates (the difference on the map is not visible to the human eye). Finally, the week number is calculated and printed in the area below the card's title, along with the dates that correspond to this week (3.13).

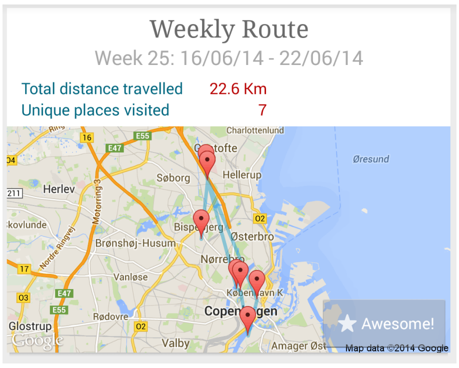

Weekly Route
Week 25: 16/06/14 - 22/06/14
Total distance travelled 22.6 Km
Unique places visited 7

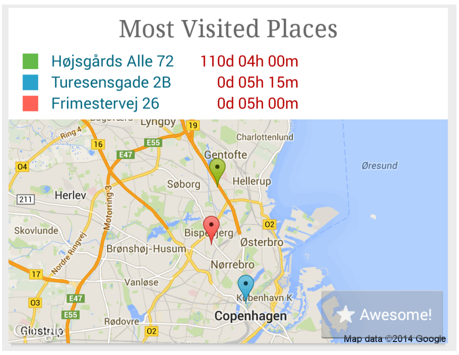
Figure 3.13: The "Weekly Route" card

### Most Visited Places

The last statistics-card is called "Most Visited Places". Its title is self-explanatory: It informs the user about the places that he/she visited the most. The stop-location data are fetched from the application's cache and the total time of visitation to each stop-location is calculated. The three most visited stop-locations along with their durations are then extracted, their addresses are fetched using Google Maps Geolocation API and they are outputted in the upper section of the card, as indicated in Figure 3.14. The durations are measured in a "days - hours - minutes" manner. The static-map that accompanies the card contains three markers pointing to the three most visited locations. Finally, the colouring of the markers corresponds to the colouring of the three rectangle boxes next to each location's address, for reasons of distinguishability (Figure 3.14).

### Continuous Feed

The six preceding paragraphs presented the six basic cards (three facts-cards and three statistics-cards) that are created every time the application is launched, or brought in the foreground. The cards are added to a scrollable list. However, a common usage scenario would be the ability to scroll further down the list in order to retrieve previous cards that are not currently available in the list. That being the case, it was decided that a *continuous feed* of cards should be implemented instead of creating just the six basic cards containing the latest information. There are many mobile applications out there that have adopted a continuous feed list implementation



Most Visited Places

| | | |
|---|---|---|
| 🟩 | Højsgårds Alle 72 | 110d 04h 00m |
| 🟦 | Turesensgade 2B | 0d 05h 15m |
| 🟥 | Frimestervej 26 | 0d 05h 00m |

Figure 3.14: The “Most Visited Places” card

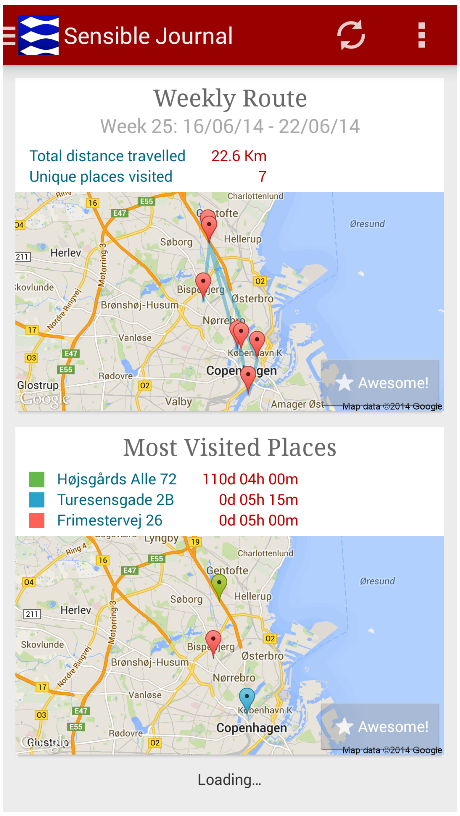
(*Google+*[17], *Facebook*[18], *Twitter*[19]). The idea is that, when the user scrolls to the end of the card list, a “Loading...” message appears to the footer of the list and the older cards are created and rendered (Figure 3.15).

The cards that appear in the continuous feed are the user’s older “Daily Route” cards. As depicted in Figure 3.15, these cards contain an “Archived” indicator, in order to be distinguished from the latest ones. Every time the list is scrolled towards the end, the previous day’s stop-location data are fetched from the cache (by using the incremental day counter field) and the corresponding card is created. Consequently, this feature provides the users with the ability to scroll down and obtain information about their past movements in a modern and user-friendly way. Due to the feel of a continuous feed, the application’s card list will be referred to as *card-feed* for the rest of this thesis.

## Detailed Views

As mentioned earlier in this chapter, each card of SensibleJournal 2014 corresponds to a more detailed visualization (detailed view), which offers additional information, along with an interactive map pinpointing the proper locations. In order to enter a detailed view, the user has to tap on one of the cards in the main card-feed. A new Android activity is launched and the information appear on screen. The user can go back to the card-feed view, at the same scrolled position, by tapping the back button. Each detailed view is built around an interactive map, which demonstrates the same stop-locations and itineraries, using lines, markers and custom map *info-windows*. An info-window is a pop-up window carrying text or images, that appears above the map when clicking on a marker[20]. Some of the detailed views offer two

[17]https://play.google.com/store/apps/details?id=com.google.android.apps.plus
[18]https://play.google.com/store/apps/details?id=com.facebook.katana
[19]https://play.google.com/store/apps/details?id=com.twitter.android
[20]https://developers.google.com/maps/documentation/android/infowindows


Sensible Journal

Weekly Route

Week 25: 16/06/14 - 22/06/14

| | |
|---|---|
| Total distance travelled | 22.6 Km |
| Unique places visited | 7 |

Most Visited Places

| | | |
|---|---|---|
| 🟩 | Højsgårds Alle 72 | 110d 04h 00m |
| 🟦 | Turesensgade 2B | 0d 05h 15m |
| 🟥 | Frimestervej 26 | 0d 05h 00m |

Figure 3.15: Feed loading

additional functionalities:

- A “details” button, which lowers a drop-down window with further details about the user’s movements and statistics.
- A “play” button, which launches an animated visualization of the data displayed on the map.

An example of a detailed view is illustrated in Figure 3.16. The “play” and “details” buttons are visible in the lower left part of the screen. The buttons on the lower right part are automatically generated by the maps’ API and provide the ability to zooming in and out. Each time a detailed view is launched, the map zooms above the rectangle that is formed by the largest and smallest latitude and longitude values, as shown in the figure (grey rectangle). This subsection will analyse all the detailed views, as well as the way their animations are implemented.

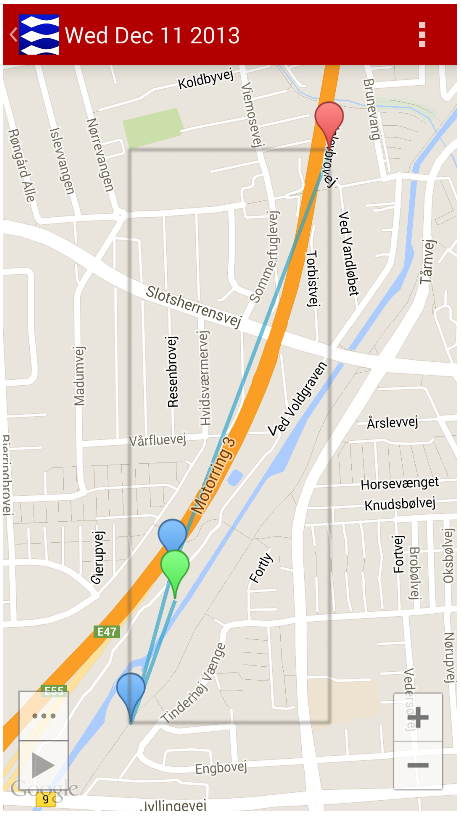


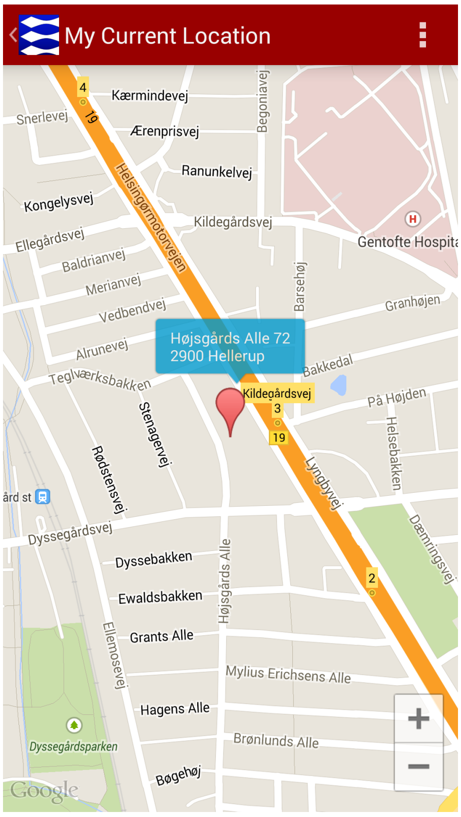
Figure 3.16: Detailed View of a card

## My Current Location - Last Visited Place

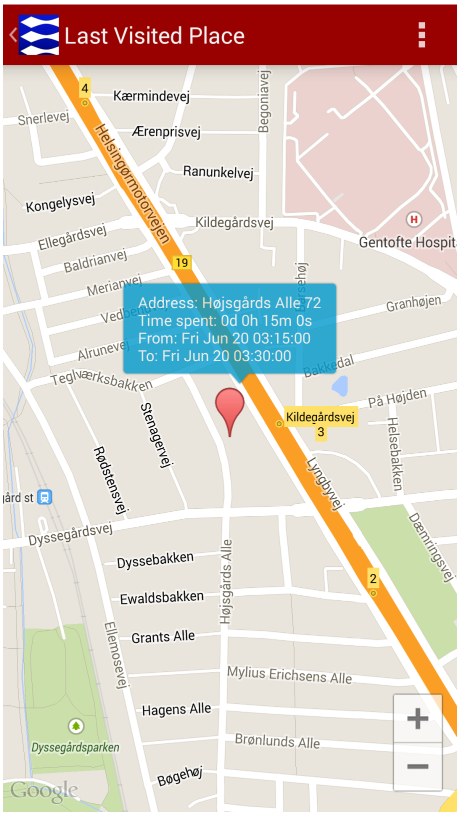
The detailed views of the first two facts cards, the "My Current Location" card and the "Last Visited Place" card, are the application's simplest ones. As shown in Figure 3.17(a) and Figure 3.17(b), they both utilize a marker on a map on the appropriate location, along with an info-window showing further information. For the "My Current Location" detailed view, the info-window carries address information, while the "Last Visited Place" one also reports the time that the user spent, along with the time he/she arrived and departed from that location. The type of each detailed view is placed on the *action bar* of the application, at the top of the screen. The user can move around the map by tapping and scrolling, or zoom in and out by double-tapping, pinching or tapping on the zoom buttons on the lower right part of the screen.

## Latest Journey - Daily/Weekly Route

This paragraph covers the next three detailed views: The "Latest Journey", the "Daily Route" and the "Weekly Route" detailed view. They are described together due to the



((a)) "My Current Location"



Address: Højsgårds Alle 72
Time spent: 0d 0h 15m 0s
From: Fri Jun 20 03:15:00
To: Fri Jun 20 03:30:00

((b)) "Last Visited Place"

Figure 3.17: Detailed Views

fact that the information that they contain, as well as the way they operate are very similar. They use markers connected by lines on the map in order to indicate the route followed by the user. In addition, they all have the aforementioned "details" and "play" buttons enabled, in order to either lower the drop-down details window, or animate according to the user's itinerary respectively. When the animation is initiated, the camera zooms to the start location (indicated by the green marker), rotates and then moves towards the next location, until it reaches the final one. While moving from one stop-location to another, their info-windows are shown, providing more detailed information about the user's arrival and departure. The way the animation is implemented will be described in a later paragraph. The info-windows are also shown by clicking on a marker on the map. As with the previous detailed views, the user can interact with the map by zooming in or out or moving it.

Figure 3.18 shows the "Latest Journey" detailed view with the details window dropped. As with the corresponding card, the average speed, distance and means of transport are indicated. Also, a list with the start and final locations' addresses and visitation durations is available. Clicking on an item in the list, zooms-in on the corresponding location and opens its info-window.

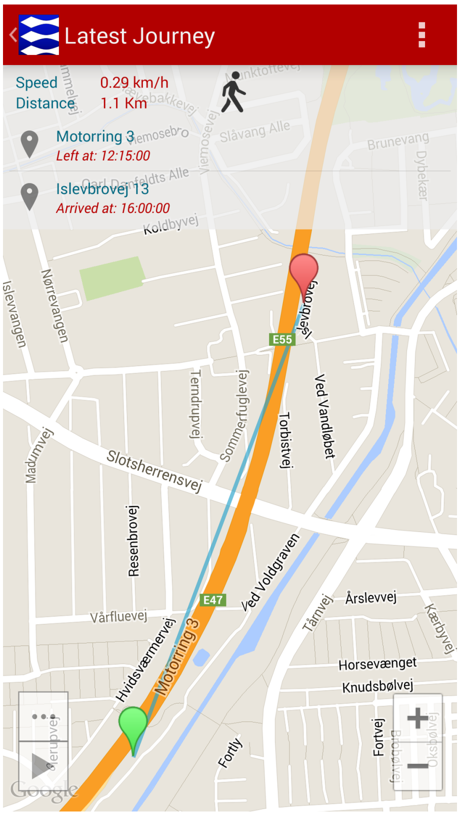
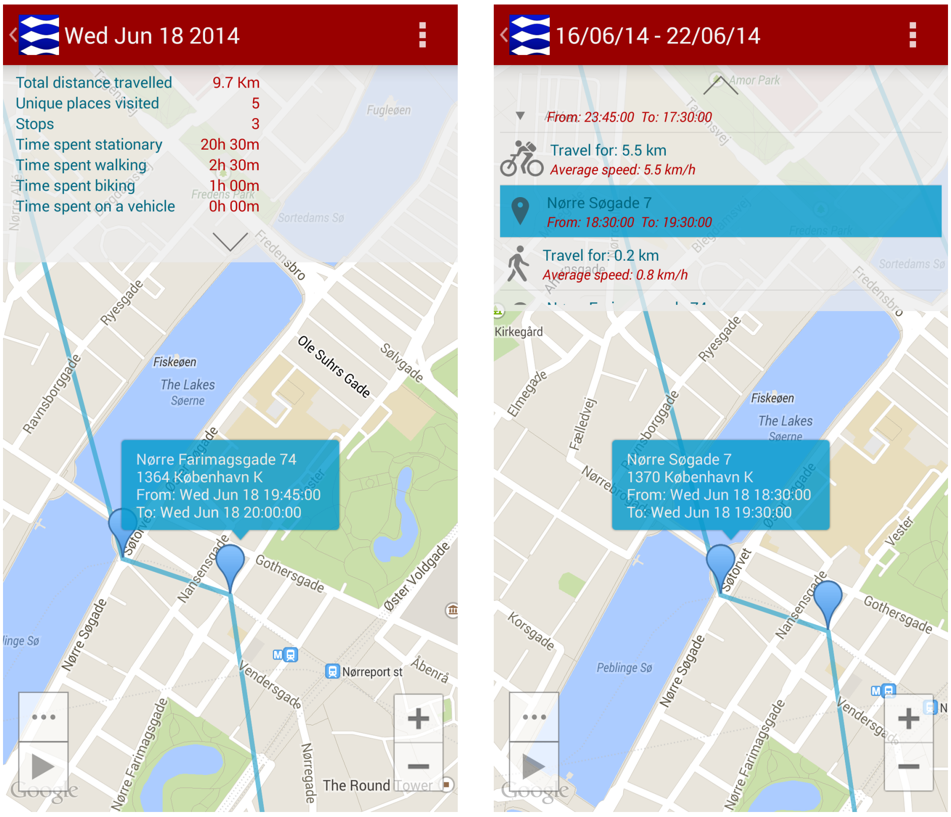


Figure 3.18: "Latest Journey" detailed view

"Daily Route" and "Weekly Route" are almost exactly the same in terms of layout. The only difference is on the data they present (daily or weekly routes). Their drop-down details window has two pages:

- A page with details about the day's/week's movements (Figure 3.19(a)) with the following entries:
  - Total distance travelled (also appears on the corresponding card)
  - Unique places visited (also appears on the corresponding card)
  - Number of stops ("Daily Route" only, also appears on the corresponding card)


Wed Jun 18 2014

| | |
|---|---|
| Total distance travelled | 9.7 Km |
| Unique places visited | 5 |
| Stops | 3 |
| Time spent stationary | 20h 30m |
| Time spent walking | 2h 30m |
| Time spent biking | 1h 00m |
| Time spent on a vehicle | 0h 00m |

Nørre Farimagsgade 74
1364 København K
From: Wed Jun 18 19:45:00
To: Wed Jun 18 20:00:00

16/06/14 - 22/06/14

From: 23:45:00 To: 17:30:00

Travel for: 5.5 km
Average speed: 5.5 km/h

Nørre Søgade 7
From: 18:30:00 To: 19:30:00

Travel for: 0.2 km
Average speed: 0.8 km/h

Nørre Søgade 7
1370 København K
From: Wed Jun 18 18:30:00
To: Wed Jun 18 19:30:00

((a)) "Daily Route" ((b)) "Weekly Route"

Figure 3.19: Detailed Views

- Time spent stationary
- Time spent walking
- Time spent biking
- Time spent on a vehicle

• A page containing a list of the route's stop-locations, separated by elements that provide information about the means of transport, distance and average speed used to commute from one location to another (Figure 3.19(b))

Lastly, it should be noticed that, when the application launches the "Daily Route" detailed view, the chosen date is printed on the action bar of the application (3.19(a)), while in "Weekly Route" detailed view, the chosen week's starting and ending date are shown 3.19(b).

**Most Visited Places**

The sixth and last detailed view of SensibleJournal 2014 corresponds to the "Most Visited Places" statistics-card. As already discussed, this card reports the user's most visited stop-locations along with the total duration of his/her visitation to each location. Whilst the card only illustrates the three most visited places, in the detailed view, the user has the ability to add more locations to the list. This feature can be accessed by clicking on the "details" button, in order to lower the drop-down window which contains the list of the most visited locations. As portrayed in Figure 3.20, two arrows are placed on the top of the drop-down window along with a number, indicating the currently visible most visited locations on the map. The user can add or remove locations by tapping on the up or down arrows respectively. Each list item contains the address of each stop-location along with the total visit duration. As in the previous detailed views, there is the ability of moving around the map and zooming in and out. Also, tapping on a marker will reveal the corresponding information in an info-window. Finally, this detailed view also provides an "animation" button. In contrast to the previous detailed views' animations, this just zooms-in at each location, reveals its info-window and then moves to the next location until the end of the list.

It is worth noticing that SensibleJournal 2014 was deigned in a way that supports easy expandability. The way the cards and the corresponding detailed views are organized, facilitates the future addition of more visualizations, as will be discussed in a later chapter. As a side-note, it should be mentioned that all the application's screenshots used in this dissertation are taken using fake test-data, for reasons of privacy. This also explains the peculiarity in many time-based values.

**Map Animation**

The preceding paragraphs presented the way the detailed views of SensibleJournal 2014 are designed and how they operate. Four of them supported a map animation which is responsible for visualizing the stop-locations of a specific route, in a sequential manner. The animation is fired-up through a "play" button at the bottom-left part of the detailed view screen. The way the animation operates is based on a Google Maps API V2 tutorial[21]:

1. Upon tapping the "play" button, the map camera zooms-in to the first stop-location.
2. When zooming ends, the heading angle towards the next location is calculated.
3. The map camera rotates according to the heading angle.
4. When rotation ends, the map camera starts moving towards the next stop-location.

[21]http://ddewaele.github.io/GoogleMapsV2WithActionBarSherlock/part3

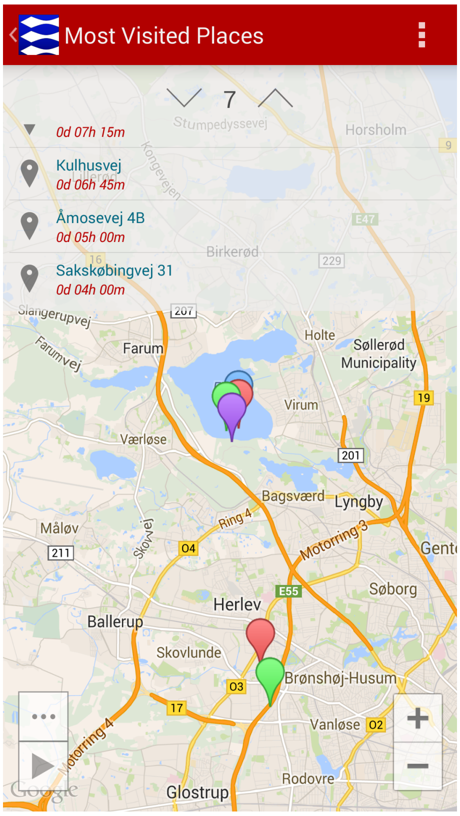


Figure 3.20: “Most Visited Places” detailed view

This procedure is repeated until the end of the location list (in case of the “Most Visited Places” detailed view, the camera does not rotate, it just zooms in and out again before moving to the next location).

The user has the ability to stop the animation: When the animation is initiated, the “play” button is altered to a “stop” button. Upon tapping on it, the animation is ceased and the camera zooms out to the initial rectangle formed by the maximum and minimum latitude and longitude in the location list. Finally, upon moving from one location to another, the corresponding info window is shown. However, if the details window is dropped during the animation all the info-windows are hidden. An example of an animation is depicted in Figure 3.21: the camera has rotated towards the direction of the next point and the corresponding info-window is shown.

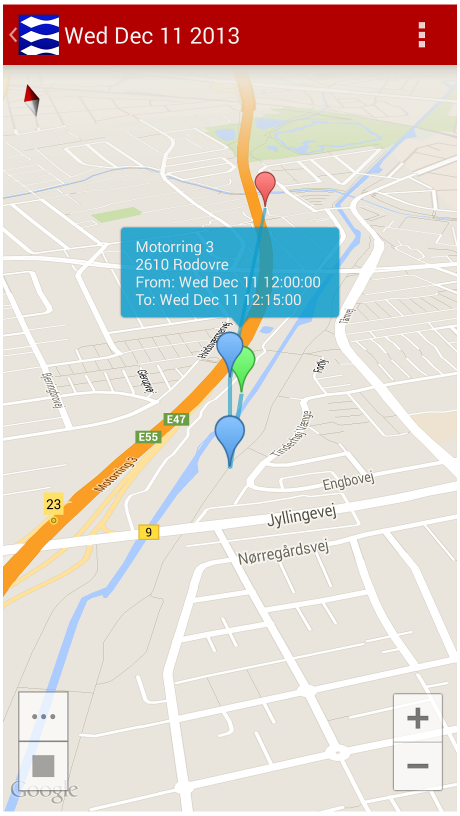
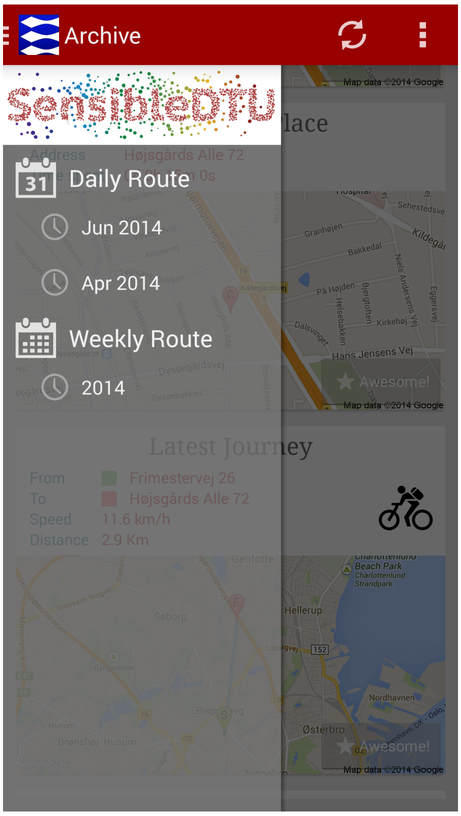


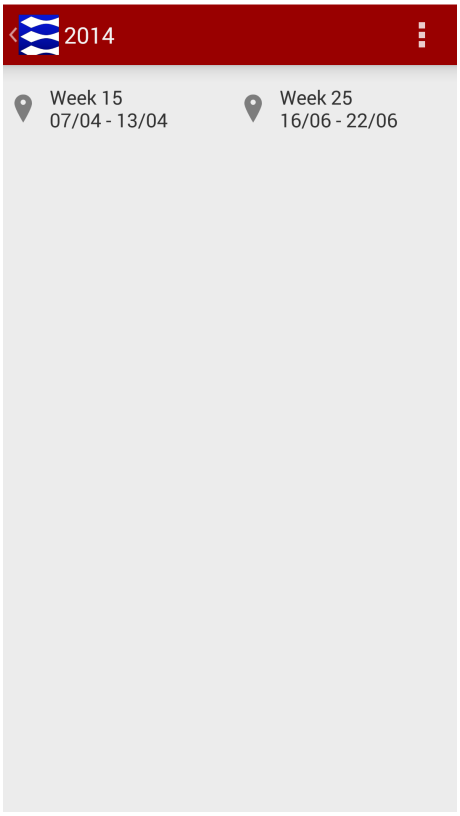
Figure 3.21: Map animation

## Archive

As discussed in Subsection 3.1, the application was designed in order to provide with a way for the user to easily access his/her past movements. This functionality is provided by the *archive*, which is accessible through the *navigation bar*, as shown in Figure 3.22(a). It can be easily drawn by swiping to the right anywhere in the main view of SensibleJournal 2014, or clicking on the title in the action bar. The archive, similarly to the continuous card feed, provides with previous daily and weekly itineraries. They are available through an expandable list and are separated by month or year respectively. Clicking on a certain month or year from the archive, launches an activity with the corresponding available daily or weekly routes (Figure 3.22(b)). Ultimately, clicking on an archive entry, launches the relevant detailed view, where the user can be informed about his/her movements and visitations during that day or week.



((a)) “The navigation drawer”



((b)) “Archive for weekly itineraries”

Figure 3.22: The archive

## Notifications

In order for the user to be reminded to check his/her personal mobility journal, the application has been empowered with a reminder sub-system, which is responsible for creating an Android notification using the native Android’s alarm manager (similarly to the data-fetching sub-system, described in Paragraph 3.2). It is a common case that notifications in smartphones can usually become intrusive, engaging disturbing sounds or vibration, or being repeated annoyingly though the day. Considering the fact that a personal informatics/life-logging application wouldn’t be of vital importance for the user’s everyday routine, it was decided not to include any sounds or vibrations in the notification, which fires-up once every three days, at 12:00 pm. The choice was made considering that the user will probably have his/her device nearby at that time. Figure 3.23 illustrates the notification: The icon is chosen to be 24x24 dp, according to the Android design principles[22]. Tapping on the notification causes the application to be launched, if closed, or brought to the foreground. If a communication with the server has occurred in the background, a loading window will appear and the card-feed will be re-rendered using the fresh data.

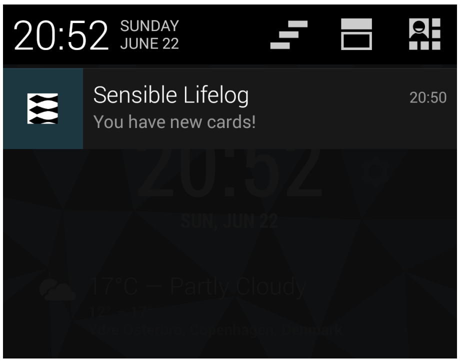


Figure 3.23: The notification

## Usage Statistics

For experimental reasons, it is crucial to acquire knowledge about how people use the application. As mentioned, the application was deployed in Google Play Store, in order to replace the older version, already downloaded and used by a number of experiment participants. Thus, it has been empowered with the ability to gather data about the way it is being used. For this to be carried out, the application creates a second SQLite database, which logs the usage data. As with the cache database, it is created the first time the application is launched. It logs events along with timestamps, created while the application is in the foreground.

In Subsection 3.2, an “Awesome!” button was referenced, appearing on every card. When clicked, the button logs the event (named after the card type) in the appropriate database along with its timestamp. Similarly, when each application activity is launched (main application activity, archive activities, detailed view activities), a current timestamp is taken and the event is stored in the usage log database.

In order to obtain the usage data from each participant, a sub-system similar to the data-fetching (REST client) sub-system, described in Paragraph 3.2 was implemented which uploads, through an SSL connection, the data to the SensibleDTU server. It also is fired up using Android’s native alarm manager and the upload is set to occur once per day from 17:00 to 20:00. Each usage log upload consists of three fields:

- *bearer_token*: The user’s unique token

[22]https://developer.android.com/design/index.html

- *appid*: The application's identifier
- *events*: Array of timestamp-event pairs, e.g. [[123456000, "MOST_VISITED"], [123456000, "MAIN"]]

The event types that are being uploaded to the server along with each user's token are:

- Entering the Main view of the application
- The application is paused or stopped
- Entering the "Daily Route" archive
- Entering the "Weekly Route" archive
- Entering the "Current Location" detailed view
- Entering the "Last Place Visited" detailed view
- Entering the "Latest Journey" detailed view
- Entering the "Daily Route" detailed view
- Entering the "Weekly Route" detailed view
- Entering the "Most Visited Places" detailed view
- "Current Location" card "Awesome!" button tapped
- "Last Place Visited" card "Awesome!" button tapped
- "Latest Journey" card "Awesome!" button tapped
- "Daily Route" card "Awesome!" button tapped
- "Weekly Route" card "Awesome!" button tapped
- "Most Visited Places" card "Awesome!" button tapped

The usage data that were received after the deployment of the application will be further examined in the next chapter, in the *Usage Analysis* subsection.

## 3.3 Usability Testing

An increasingly important part of a mobile (or any other) application process is *usability testing*. It is a testing procedure, involving real users or specialists/researchers in order to evaluate the design and/or functionality of the application during, or after the design and implementation process. According to [Gon09], usability test "is a process of systematically collecting the usability data of the interface, assessing and improving the data". Such an evaluation during the design process is called *formative evaluation*, while a usability test after the implementation of the application is called *summative evaluation* [Lar13]. Usability tests can hide surprisingly beneficial results and conclusions about the final design and functionality of the application. For instance, proper usability testing can help achieve the desired functionality of the application in a more convenient manner and in less time (reducing the development's expenditure), as well as improve the overall usage quality and thus establish it as more competitive in the market.

There are three usability test methods [Gon09]:

- *Virtual Test Method*: The role of the users is played by specialists or researchers in order to get usability feedback.
- *User Examine Method*: Record or observe real users operating the application and retrieve real-time feedback.
- *Experimental Comparison Method*: Apply experimental methods on different versions of the design or functionality aspect and decide according to the results.

The two first methods were adopted during the development of SensibleJournal 2014. Virtual Test Method was used during the design and implementation of the application (formative evaluation), where the tester's role was played by the thesis supervisor and assistant supervisors. User Examine Method was used after the completion of development (summative evaluation); a pre-release version of the application was handed-out to three testers who owned a compatible device in order to use it in real life. In both the evaluation periods, the usability feedback was taken in real time, verbally, using the *thinking aloud* method. Thinking aloud is a vastly used, simple and effective usability testing method, which involves verbal communication between the testers and the designer: While operating the application, the testers verbally express their opinions and thoughts in real-time. The developer listens and takes notes or records them, in order to retrieve useful feedback. According to Jacob Nielsen [SHM10], "In a thinking aloud test, you ask test participants to use the system while continuously thinking out loud — that is, simply verbalizing their thoughts as they move through the user interface". [Nie12] demonstrates an interesting research in which the researchers used a thinking aloud method in order to acquire feedback about two airline websites. The results shown that many users agreed to pay a more expensive ticket through the websites, due to possible website deception. The paper concludes that, such evaluation methods should be standardized in the future, in order to avoid similar situations.

Jacob Nielsen, in [SHM10] describes thinking aloud method as a “window on the soul, letting you discover what users really think about your design”. He summarizes the benefits of the method in the following points:

- It’s *cheap*, because it is verbal and the only requirements could be a notepad or a recording device.
- It’s *robust*, due to the fact that it is simple; more sophisticated usability methods with complex methodologies could include mistakes, thus producing more misleading feedback.
- It’s *flexible*, as it can be used easily at any time during the development of an application.
- It’s *convincing*, because in the end only the user’s opinion is what matters and this method is completely dependent on user’s testimonies.
- It’s *easy to learn*, again because it only involves the tester using the application while speaking his/her thoughts and the designer taking notes.

During the design and implementation process of SensibleJournal 2014, both formative and summative evaluation was performed using the thinking aloud method. Initially, an introduction to the application and the way it operates, or (in case of formative usability testing), a presentation of new changes since the previous session, was demonstrated to the testers. Throughout the testing process, notes were obtained while listening to the feedback, using either a pen and paper, or a note-taking mobile application. Upon finishing the session, several questions regarding the overall quality of the application were posed, such as:

- Did you understand the purpose of the application?
- Would you use it in a daily manner?
- Does the design follow today’s trends?
- Was the application fast and easy to use?

The above more generic questions helped in justifying the notes taken during the usability testing session and making improvements that might not have been suggested, such as increasing the overall responsiveness during activity transitions or animations. In general, the reaction of the testers was positive, showing an eagerness for using the final version. Following the usability testing sessions, the notes takes were evaluated and actions to be taken were decided. To sum up, the aspects of the application that changed due to either formative or summative feedback, are set out below:

- During prototyping, it was decided that a card filter will be available in navigation drawer. It was suggested for it to be replaced with the "Archive" both for reasons of accessibility as well as redundancy, due to the fact that a card filter was not determined as really useful (formative).
- Initially there was no card-feed implemented in SensibleJournal 2014. It was included after a specific usability test session (formative).
- The layout of the drop-down window in detailed views was also a result of usability testing (formative).
- The colors of the application were also changed several times during design (formative).
- Post-development, it was proposed that the tutorial cards should have a different color from the functional ones (summative).
- "Loading" messages were introduced whenever there is a more time consuming transition between several views of the application (summative).
- The testers also proposed changes in the descriptive strings of the tutorial cards (summative).
- Initially the application used the word "itinerary" for indicating the user's daily and weekly journeys. It was changed to the more informal "route" (summative).
- Tutorial cards were confusing due to not being clickable. An Android native *toast* message encouraging to swipe them away was later added upon clicking on them (summative).
- Initially, "Weekly Itinerary" and "Daily Itinerary" was outputted as titles for the daily and weekly route activities. After a relevant proposal, they were changed to indicate the exact day and week number respectively (summative).

CHAPTER 4

# Experiment

This chapter describes the experimentation phase of this thesis, which consists of the deployment of SensibleJournal 2014 in Google Play Store, in order for the SensibleDTU participants to install or update the previous version on their devices (Section 4.1) and the further analysis of the usage data gathered from each application instance (Section 4.2).

## 4.1 Deployment

After having completed the development and usability testing phases, SensibleJournal 2014 had to be released to public, for the current participants to download. According to the *Android Launch Checklist*[1], it is significantly valuable for a pre-release version of the application to be distributed to several trusted beta-testers, in order to confirm the normality of its installation/update and detect any last-time bugs. After a small group of beta testers was formed, the beta version was distributed to them for testing. In order to receive the update to the 2014 version of SensibleJournal, the testers had to:

- Have SensibleDTU data collector installed.
- Have SensibleJournal 2013 installed.
- Have logged-in in both.
- Have accepted the invitation encouraging them to become beta testers of the application.
- Be members of the SensibleJournal 2014 beta community on Google+.

Were the above satisfied, a notification from Google for updating the application to the new version was sent to each beta-tester after several hours. During the beta-testing phase, a bug considering the different devices' locales was discovered, which was causing the application to crash on devices with the Danish locale enabled.

With the beta-testing phase completed, the application was finally launched on the Google Play Store for general consumption, along with a description and several screen-shots. A total number of 417 participants installed (or upgraded from the 2013 version) and used it. The experimentation phase's duration was four weeks.

[1]http://developer.android.com/distribute/tools/launch-checklist.html

## 4.2 Usage Analysis

SensibleJournal 2014 was set to upload usage data from each participant's phone to the SensibleDTU server every afternoon from 17:00 to 20:00, starting from the next day after the first launch of the application. The access to the usage database on the server is done through a *phpMyAdmin*[2] session, which allows access to remote databases through an internet browser. The usage entries are extracted from the database and downloaded in JSON format (also, many other format options are available for download). An example of a user's usage data is shown below:

```
{
    "id": 1141,
    "timestamp": "2014-06-29 12:23:56",
    "user": "6d0d837492d3a5f6cc54978a40229f",
    "appid": "sensiblejournal",
    "event": "MAIN",
    "uniqueness_hash": null
},
{
    "id": 1142,
    "timestamp": "2014-06-29 12:25:13",
    "user": "6d0d837492d3a5f6cc54978a40229f",
    "appid": "sensiblejournal",
    "event": "DAILY_ITIN",
    "uniqueness_hash": null
},
{
    "id": 1143,
    "timestamp": "2014-06-29 12:26:04",
    "user": "6d0d837492d3a5f6cc54978a40229f",
    "appid": "sensiblejournal",
    "event": "MAIN",
    "uniqueness_hash": null
}
```

Listing 4.1: Example of usage data as downloaded from the server

As shown in the illustrated example, each usage database entry is consisted of six different fields, characterizing the event that took place. The first field, "id", is an auto-incremented identifier for every entry and is followed by "timestamp", which indicates the exact date and time of the event. A unique hexadecimal user identification number is also provided ("user" field), as well as the application's string identifier. Finally, each event's category is denoted by the "event" attribute. There is also a "uniqueness_hash" field, intended for ensuring the uniqueness of each entry through a hashing algorithm, but is not used in this work.

Four weeks after the release of SensibleJournal 2014, the usage data were extracted from SensibleDTU server and several *Python* scripts were developed, in order to anal-

[2] http://www.phpmyadmin.net/

yse and provide insights about the way the experiment participants used the application and which aspects of it they found to be more useful. The usage data analysis, along with several observations are further described in following sub-sections.

### Card “Awesome!” Clicks

Figure 4.1 shows the total number of the “Awesome!” button clicks during the experimentation phase. It was created by counting the corresponding events gathered from the usage data. It is apparent that the “Daily Route” card received the most attention, which leads to the conclusion that people prefer to mostly self-reflect on the daily summary of their movements. “Current Location” and “Last Visited Place” cards also received a significant amount of “Awesome!” taps, however, this can be considered as biased, due to the fact that they appear first on the list and the “Awesome!” button could just be pressed several times out of curiosity. The least favoured card was “Latest Journey”. It should be noted, at this point, that during the period that the application was being used by the participants, the stop-location API, being still in development, had a several-hour delay for producing results based on the uploaded information. Hence, data uploaded for example on Tuesday could appear on SensibleJournal 2014 on Thursday morning. This could be one of the reasons that “Latest Journey” received fewer “Awesome!” taps. A future, faster version of the API could aid increasing this card’s interest.

### Detailed Views Preferences

Figure 4.2 illustrates the total time spent daily on each card’s corresponding detailed view. The durations were calculated using the timestamps of the events in the following way: Considering two consecutive events, e.g. with “DAILY_ITIN” and “MAIN” (see Listing 4.1), the total time spent on the activity represented by the first one, is the time difference between its timestamp and the “MAIN” event’s timestamp. In order to avoid taking into account durations of false events, all the possible valid transitions between the application’s views were determined. Possible invalid events that could be created by any reason were discarded. For example, all the possible valid transitions from the card-feed view are:

- From card-feed to Current Location detailed view
- From card-feed to Last Place Visited detailed view
- From card-feed to Latest Journey detailed view
- From card-feed to Daily Route detailed view
- From card-feed to Weekly Route detailed view
- From card-feed to Most Visited Places detailed view

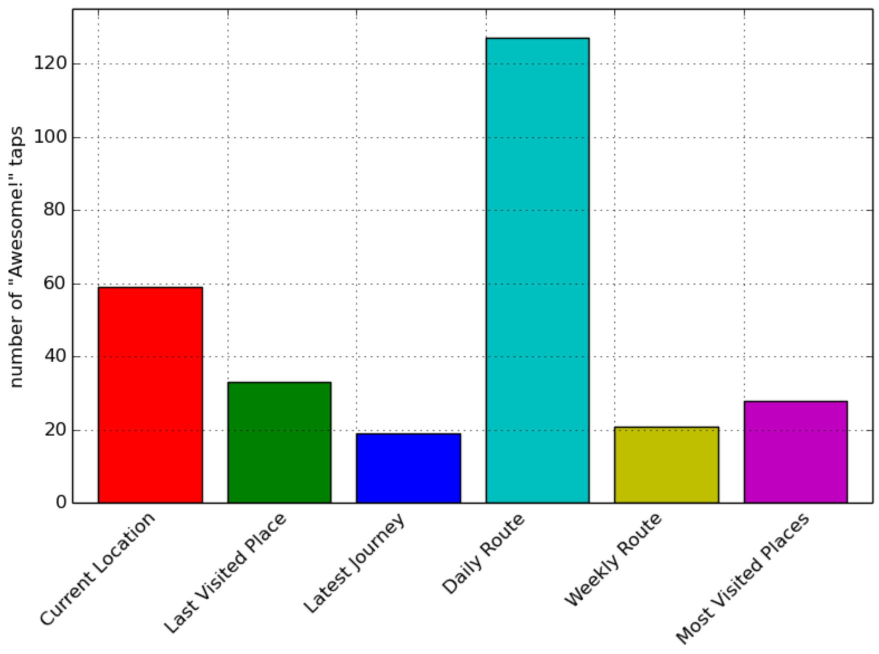


Figure 4.1: Number of “Awesome!” clicks for each card

- From card-feed to Daily Archive view
- From card-feed to Weekly Archive view
- From card-feed to paused state

It is apparent (Figure 4.2) that the most-used detailed view was the “Daily Route”. This confirms the previous observation, that people prefer to self reflect on their daily mobility information. “Weekly Route” is also visited for a significant amount of time, especially during weekends, or at the beginning of a week, indicating that the users preferred to reflect on the passed week’s movement summary mostly upon its ending. It’s worth noticing that, “Daily Route” and “Weekly Route” cards-detailed views are the ones that provide details on a person’s mobility history and are available both through the card-feed and the archive of the application. This, partially justifies the fact that they receive the most attention, but also suggests that people spend more time reflecting on older and summarized personal mobility data (e.g. for reasons of recollecting or reminiscing). However, that does not indicate that they tend to ignore information about their more recent movements: The “Current Location” detailed view was visited almost every day, as shown in Figure 4.2. Hence, it would be safe to assume that people also tend to self-reflect on their latest mobility information,

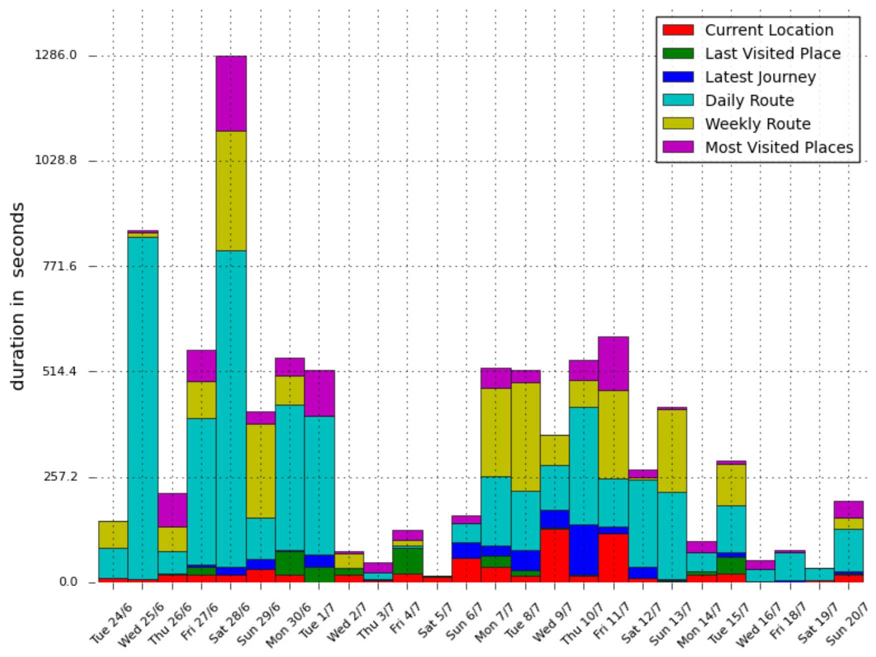


Figure 4.2: Time spent in detailed views per day

but do not spend larger amounts of time on it; they rather just "check and exit". Conclusively, "Last Visited Place" and "Latest Journey" were the least visited detailed views, possibly due to the aforementioned stop-location API delay.

### Usage Frequency

Figure 4.3 shows the daily usage frequency of SensibleJournal 2014, which is the total number of times that the application was launched or resumed during each day. This number is calculated by counting the number of valid pause events. This can be realized by the fact that each pause signifies a previous launch or resume. Initially, a rather small amount of users installed it, which could be the result of two elements: Firstly, a lower usage ratio during the first days after a release can always be expected, due to the fact that not all the participants have yet installed the application. Secondly, in the case of SensibleJournal 2014, a bug was detected during the first two days after the release which caused the application to crash in several devices. This was immediately corrected and on the third day a significant increase in usage was noticed. However it is apparent that the usage frequency is slightly decreasing, which again, is an expected behaviour, considering that a user might use an application more, right after he/she installed it, for reasons of gaining knowledge about the way it operates. Apart from the initial days after deployment, the highest number of launches is noticed again during weekends, or at the beginning of a week, confirming the previously observation that users tend to self-reflect on the passed week's movement summary mostly upon its ending.

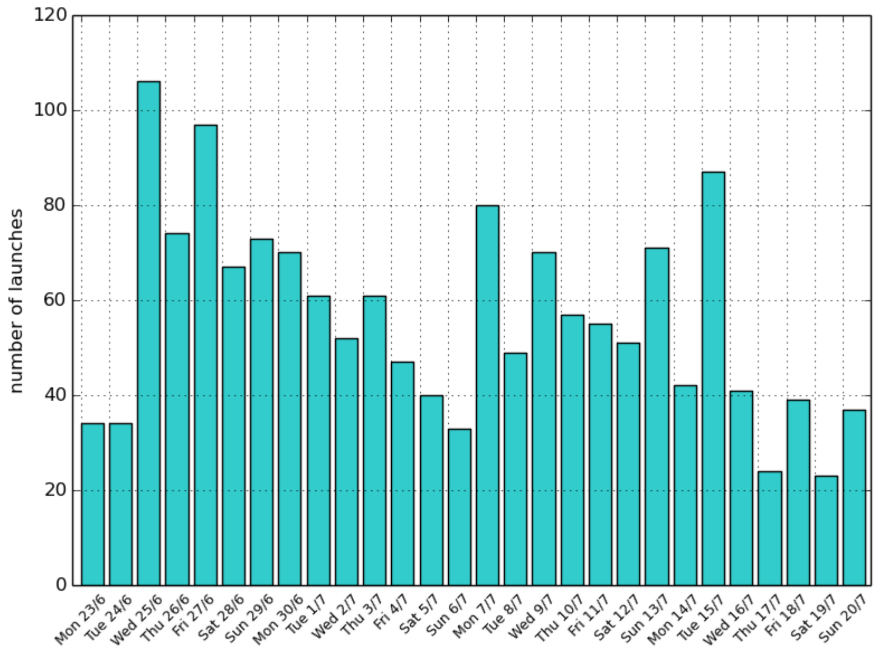


Figure 4.3: Frequency of usage per day

## Daily Active Users

Finally, Figure 4.4 depicts information about each day's total active users. Shortly after the release (Monday 23/6 and Tuesday 24/6), a smaller number of users launched the application (about 12-15), for the same two reasons as the ones that caused a reduced amount of launches during the first two days, described in the previous subsection. In general, Figure 4.4 confirms all the above mentioned remarks, such as an increased interest during the weekends/first weekdays and a slight decrease of activity over time. As mentioned, a total of 417 experiment participants installed or upgraded SensibleJournal to the 2014 version. A mean value of ≈ 39 people launched the application every day, which is a 12.3% of all the users. This suggests that people do not tend to launch the application daily, but rather a few times per week, or per larger interval.

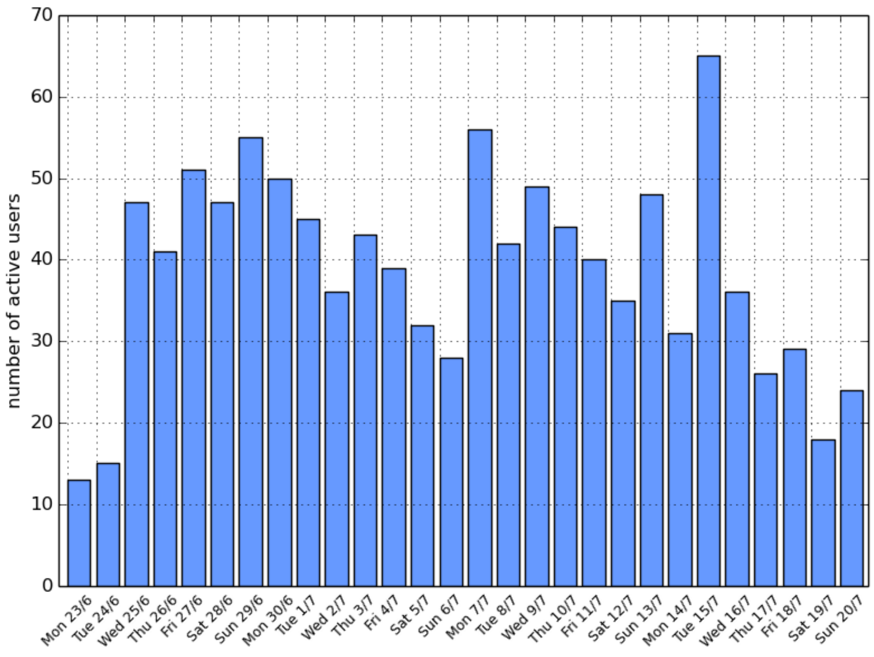


Figure 4.4: Number of active users per day

CHAPTER 5

# Discussion

This chapter contains a general discussion, concerning specific insights about the experiment's results (Section 5.1) introduced at the previous chapter and the way they could be exploited for improving the application, followed by an attempt to determine the placement of SensibleJournal 2014 according to the theoretical aspects presented in Chapter 2, in conjunction with possible future work (Section 5.2).

## 5.1 Results - Future Work

Chapter 4 presented several usage data statistics, emerged from the elaboration of the usage information gathered from each participant's smartphone. In particular, the analysis involved:

- *Card "Awesome!" Clicks*: The number of "Awesome!" button taps for each card.
- *Detailed Views Preferences*: The total time people spent in each detailed view every day.
- *Usage Frequency*: How many times the application was launched daily.
- *Daily Active Users*: How many distinct users launched the application every day.

A diagram was demonstrated for each of the above statistical categories, followed by a discussion attempting to point out specific trends and insights that can be derived about the way SensibleJournal 2014 was used, enumerated below:

- People generally prefer to reflect on the daily summary of their movements.
- The sequence of the cards may influence the process of self-reflection.
- People mostly reflect on their weekly mobility summary during weekends or first days of a week.
- People spent more time reflecting on their past movements (recollecting/reminiscing).
- People are interested in their latest/current mobility information more frequently.

- General interest about the application is reduced as the time passes by.
- People do not launch the application every day.

These above-stated insights can be employed in order to determine possible alterations and improvements for SensibleJournal 2014. They can also be taken under consideration in developing other similar personal informatics applications. Special attention should be given on the way the daily mobility information is presented: New card/detailed view visualizations could be added, providing more self-reflecting potential, such as the detection of patterns in one's movements (e.g. using spirals [LCL13]). Furthermore, the cards could be automatically re-arranged, according to the way the application is operated. A more sophisticated notification system should encourage the users to more effectively access their mobility information, for example special reminders for checking a week's summary during the weekend, or a small report about his/her latest commute when the user arrives at a new place. This could also keep the interest at higher rates, avoiding the decrease of usage over time. Finally, special attention should be given to properly support recollecting/reminiscing, by adding, for example, the ability to capture and store pictures, along with written or audio notes for each stop-location.

It is worth noting that, the usage statistics could be more accurate and informative if they concerned more than a few weeks of usage but such an option wasn't possible due to thesis deadline regulations.

## 5.2 Theoretical Placement - Future Work

SensibleJournal 2014 can be described as a life-logging application, due to the fact that it provides an effortless way of data capture [SW10]. It is a total-capture personal informatics system, since the SensibleDTU data collecting application that runs in the background, ubiquitously gathers data from the smartphone's sensors with respect to a fixed time interval. However, the server-side elaborated stop-location data that SensibleJournal 2014 is built upon, could also be described as situation-specific, as they concern only the places that the user stays at, for more than a predefined amount of time. It is a single-layer data personal informatics system, but, due to its expandability potential it can be transformed to a multi-layer one, by adding more visualizations, for example based on social data, also gathered by the SensibleDTU project.

Prototyping and usability testing played an important role in the development of the application. Prototyping allowed for early, low-level design decisions, which provided with the proper insight for how the application should look and operate like. Through formative evaluation during the development, important decisions that helped avoiding the lead into undesired paths and results were taken. Finally, summative evaluation and usability testing, helped correct and adjust specific aspects of the application that were missed or not given the necessary significance.

In Chapter 2, the five-stage schema according to which a personal informatics system can be organized was presented [LDF10b], laying the foundations for a more layered development of such applications. As discussed, the stages can be user-driven or system driven. SensibleJournal 2014's placement according to this model is described below:

- Preparation Stage: Currently, the type of data that SensibleJournal 2014 operates on, are predetermined (GPS data). However, a user-driven procedure, such as a survey, can always be conducted, altering or adding more types of data, according to users' desires.
- Collection Stage: The collection phase of SensibleJournal 2014 is automated (system-driven) and is performed by the SensibleDTU data collecting application.
- Integration Stage: The evaluation and processing of the data is also system-driven, either server-side (stop-location data determination) or client-side (elaboration of the stop-location data in order to determine the user's movements and the relevant statistics).
- Reflection Stage: The reflection stage is supported by automatically (system-driven) generating map-based visualizations, using the collected data. Both long-term and short-term reflection are possible, as the application provides visualizations of either one's latest information (first six cards and their detailed views), or older information (archived cards and their detailed views).
- Action Stage: This stage is not supported automatically by SensibleJournal 2014 and is completely up to the user (user-driven) to determine his/her actions, based on his/her self-reflection experience. A system-driven action stage enabled version that automatically urges users to take specific actions according to their elaborated data could be an interesting future work.

As discussed in Chapter 2, according to [LDF11], another important aspect of personal informatics and life-logging systems is the kinds of questions that people ask about their personal information. The importance of providing answers to these questions becomes apparent in developing such systems, as it clarifies their goals. [LDF11] concluded in six such basic questions (status, history, goals, discrepancies, context and factors). SensibleJournal 2014 does not directly provide an answer to all of them, a fact that could lead to more future work possibilities:

- Status: SensibleJournal 2014 provides the possibility for the users to check their movement status. More specifically, the facts-cards ("My Current Location", "Last Visited Place" and "Latest Journey"), provide information about the most latest available information, allowing the status checking.
- History: SensibleJournal 2014 provides two ways for the users to access their history, in order to remember, find trends or patterns: By accessing the archive

which contains older daily and weekly movements, or by scrolling down the card list.

- Goals: By viewing their movement history and finding patterns in it, the users can set their own goals according to their data. Automatic or manual goal determination is not supported by SensibleJournal 2014, but can be an interesting future work.
- Discrepancies: As an addition to the goal determination, the application could provide with a way to visualize, compare and contrast one's current status with his/her set goals, in order to answer this question.
- Context: The answer to this question could be aided by using data from more sources. Semantic-web technologies could also be used (e.g. ontologies, as described in [SHKK10] and [LGHL10]), allowing for easier retrieving of contextual information .
- Factors: Again, the answer to this question could be aided using semantics.

Another potential enhancement area for SensibleJournal could be the attempt to support the "five Rs" [SW10], in order to aid the memory of its users. As shown below, using multilayer data is vital for moving towards that direction:

- Recollecting: Connecting stop-locations with images, gathered automatically by a wearable camera (e.g. *Google Glass* or Microsoft SenseCam), or manually by the user could help "mentally re-living" situations.
- Reminiscing: Extending the previous point, audio and video could be added to the gathered images, supporting reminiscing through automatically created visualizations.
- Retrieving: Facebook or other social media data along with the data collected by the SensibleDTU data collector, could be connected with SensibleJournal 2014's stop-locations, in order to allow retrieving specific digital information.
- Reflecting: SensibleJournal 2014's "Most Visited Places" card, provides with an insight about where the user stays the most, aiding the detection of patterns in his/her life. More algorithms for automatic movement pattern detection, along with specific visualizations could further support an individual's reflection on his/her personal data.
- Remembering intentions: Using data from various sources, along with proper visualizations could be proven helpful for remembering one's intentions. This could also be aided by using semantic-web technologies, thus providing the necessary contextual information needed to enhance the user's memory.

Finally, the following future work proposals could be of a particular interest:

- Addition of cards that visualize the usage of the application, allowing the user to self-reflect on the way he/she uses SensibleJournal.
- SensibleDTU data collector could gather data from several wearable computer devices connected to the phone (e.g. heart rate sensors, cameras, etc.), so that they could be used to create new cards and detailed views.
- Ability to connect and share data with other applications (such as Facebook, Google+, Twitter, etc.), allowing the users to post images or information about their movements, achievements, etc.
- Addition of a more advanced algorithm for detecting the means of transport that the user used to travel between his/her stop-locations (e.g. Bus/Train WiFi detected networks, or a larger number of bluetooth detected devices could indicate public transportation).

# CHAPTER 6 Conclusion

This work presented a personal informatics application called SensibleJournal 2014, developed under the SensibleDTU project. The application was deployed in Google Play Store in order to replace its previous version, SensibleJournal 2013. Usage data were automatically uploaded to the SensibleDTU server, which were afterwards extracted, analysed and several insights about the way the application was used were presented. Apart from outlining the development and experimentation phase of SensibleJournal 2014, this text attempted to place it in the proper position with respect to the current academic and research work that has taken place in the domain of personal informatics. This placement, along with the simple expandability potential of the application, allowed the proposal of numerous possible future enhancements that could establish SensibleJournal 2014 as a complete and robust multi-layer personal informatics application that allows for easy and fun self-reflection on one's personal data.

APPENDIX A

# Static Maps API V2

This appendix presents a short description about the way Google's Static Maps API V2 was used, for the creation of the static mini-maps appearing on the application's cards. The API receives standard HTTP requests, containing the parameters for creating the maps. The request format that was employed throughout SensibleJournal 2014 is shown below:

```
"http://maps.googleapis.com/maps/api/staticmap?&size=510x260&+
 maptype=roadmap&sensor=false&scale=2&+
 markers=color:green%7Clabel:S%7C55.69615936279297,12.443717002868652&+
 markers=color:red%7Clabel:F%7C55.67926025390625,12.47945499420166&+
 path=color:0x29A3CC|weight:4|55.69615936279297,12.443717002868652|+
 55.67926025390625,12.47945499420166"
```

Listing A.1: Google Static Maps V2 request

The *size* attribute is the size of the image to be returned in pixels (length X width), *maptype* determines the graphical style of the map, *sensor* is used when a location sensor is available and *scale* is a multiplier of the pixels returned. The *markers* attribute adds a marker on the map with a certain color, label and coordinates, while the *path* instruction creates a coloured line between two specified points on the map. The above URL returns a 1020x520 map, containing two locations pinpointed using a green and a red marker (with “S” and “F” as labels respectively), connected via a cyan coloured line (Figure A.1).

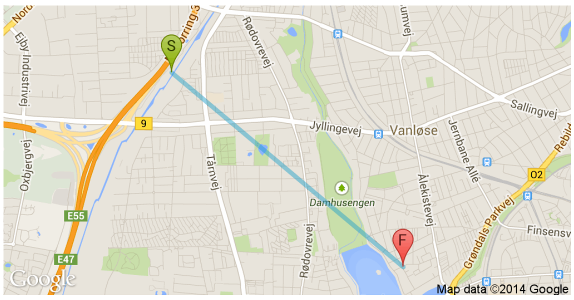


Figure A.1: Map returned by the Static Maps API V2

# Bibliography

[AEH13] Adrienne H. Andrew, Kevin Eustice, and Andy Hickl. Using location lifelogs to make meaning of food and physical activity behaviors. In *Proceedings of the 7th International Conference on Pervasive Computing Technologies for Healthcare and Workshops*, pages 408–411. ICST, 2013.

[AF11] Amin Abdalla and Andrew U. Frank. Personal geographic information management. In *Proceedings of the Workshop on Cognitive Engineering for Mobile GIS, Belfast, USA*. CEUR, 2011.

[AFH12] Masanobu Abe, Daisuke Fujioka, and Handa Hisashi. A life log collecting system supported by smartphone to model higher-level human behaviors. In *Proceedings of the 2012 Sixth International Conference on Complex, Intelligent, and Software Intensive Systems (CISIS)*, pages 665–670. IEEE, 2012.

[AH11] Jehan Alallah and Annika Hinze. Feeding the digital parrot: Capturing situational context in an augmented memory system. In *Proceedings of the 23rd Australian Computer-Human Interaction Conference*, pages 1–10. ACM, 2011.

[CGG$^+$06] Mary Czerwinski, Doug Gage, Jim Gemmell, Tiziana Catarci, Catherine C. Marshall, Manuel Perez-Quinones, and Meredith M. Skeels. Digital memories in an era of ubiquitous computing and abundant storage. *Communications of the ACM*, pages 44–50, 2006.

[CLL13] Andrea Cuttone, Sune Lehmann, and Jakob Eg Larsen. A mobile personal informatics system for visualizing mobility and social interactions. In *Proceedings of the 1st ACM international workshop on Personal data meets distributed multimedia*, pages 27–30. ACM, 2013.

[CLL14] Andrea Cuttone, Sune Lehmann, and Jakob Eg Larsen. Inferring human mobility from sparse low accuracy mobile sensing data. In *3rd ACM Workshop on Mobile Systems for Computational Social Science(MCSS 2014)*. ACM, 2014.

[CMT$^+$08] Sunny Consolvo, David W. McDonald, Tammy Toscos, Mike Y. Chen, Jon Froehlich, Beverly Harrison, Predgrag Klasnja, Anthony LaMarca, Louis LeGrand, Ryan Libby, Ian Smith, and James A. Landay. Activity sensing in the wild: A field trial of ubifit garden. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 1797–1806. ACM, 2008.

[CNBM$^+$12] Masashi Crete-Nishihata, Ronald M. Baecker, Michael Massimi, Deborah Ptak, Rachelle Campigotto, Liam D. Kaufman, Adam M. Brickman, Gary R. Turner, Joshua R. Steinerman, and Sandra E. Black. Reconstructing the past: Personal memory technologies are not just personal and not just for memory. *Human–Computer Interaction [DOI:10.1080/07370024.2012.656062]*, pages 92–123, 2012.

[CPL14] Andrea Cuttone, Michael Kai Petersen, and Jakob Eg Larsen. Four data visualization heuristics to facilitate reflection in personal informatics. In *Proceedings of the 16th International Conference on Human-Computer Interaction.* HCII, 2014.

[CSS$^+$12] Dan Cosley, Schwanda Victoria Sosik, Jonathon Schultz, S. Tejaswi Peesapati, and Soyoung Lee. Experiences with designing tools for everyday reminiscing. *Human-computer Interaction*, pages 175–198, 2012.

[DPTC$^+$12] Aiden R. Deherty, Katalin Pauly-Takacs, Niamh Caprani, Cathal Gurrin, Chirs J.A. Moulin, Noel E. O'Connor, and Alan F. Smeaton. Experiences of aiding autobiographical memory using the sensecam. *Human-computer Interaction [DOI:10.1080/07370024.2012.656050]*, pages 151–174, 2012.

[FDK$^+$09] Jon Froelich, Tawanna Dillahunt, Predgrag Klasnja, Jennifer Mankoff, and Sunny Consolvo. Ubigreen: Investigating a mobile tool for tracking and supporting green transportation habits. In *Proceedings of the 27th International Conference on Human Factors in Computing Systems*, pages 1043–1052. ACM, 2009.

[FFD12] Chloe Fan, Jodi Forlizzi, and Anind Dey. A spark of activity: Exploring informative art as visualization for physical activity. In *Proceedings of the 2012 ACM Conference on Ubiquitous Computing*, pages 81–84. ACM, 2012.

[Gon09] Chao Gong. Human-computer interaction: The usability test methods and design principles in the human-computer interface design. In *Proceedings of the 2nd IEEE International Conference on Computer Science and Information Technology*, pages 283–285. IEEE, 2009.

[JPL$^+$13] Camilla B. Falk Jensen, Michael Kai Petersen, Jakob Eg Larsen, Arkadiuzs Stopczynski, Carsten Stahlhut, Marieta Georgieva Ivanova, Tobias Andersen, and Lars Kai Hansen. Spatio temporal media components for neurofeedback. In *Proceedings of the IEEE International Conference on Multimedia and Expo Workshops (ICMEW)*, pages 1–6. IEEE, 2013.

[KK10] Judy Kay and Bob Kummerfeld. Mneme: A framework to support people in their sisyphean tasks. *School of Information Technologies, University of Sydney*, 2010.

[KLK$^+$12] Minkyung Kim, Dong-Wook Lee, Kangseok Kim, Jai-Hoon Kim, and We-Duke Cho. Predicting personal information behaviors with lifelog data. In *Proceedings of the 9th International Conference & Expo on Emerging Technologies for a Smarter World (CEWIT)*, pages 1–3. IEEE, 2012.

[KSWK10] Vaiva Kalnikaite, Abigail Sellen, Steve Whittaker, and David Kirk. Now let me see where i was: Understanding how lifelogs mediate memory. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 1–10. ACM, 2010.

[Lar13] Jacob Eg Larsen. Mobile application prototyping, lecture: Usability evaluation, 2013. Technical University of Denmark.

[LCL13] Jakob Eg Larsen, Andrea Cuttone, and Sune Lehmann. Qs spiral: Visualizing periodic quantified self data. In *Proceedings of the CHI 13 Personal Informatics in the Wild: Hacking Habits for Health & Happiness*, 2013.

[LDF10a] Ian Li, Anind Dey, and Jodi Forlizzi. Know thyself: Monitoring and reflecting on facets of one's life. In *Proceedings of Extended Abstracts on Human Factors in Computing Systems*, pages 4489–4492. ACM, 2010.

[LDF10b] Ian Li, Anind Dey, and Jodi Forlizzi. A stage-based model of personal informatics systems. In *Proeedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 557–566. ACM, 2010.

[LDF11] Ian Li, Anind Dey, and Jodi Forlizzi. Understanding my data, myself: Supporting self-reflection with ubicomp technologies. In *Proceedings of the 13th international conference on Ubiquitous computing*, pages 557–566. ACM, 2011.

[LGF$^+$13] Ian Li, Catherine Grevet, Jon Froehlich, Ernesto Ramirez, and Jakob Eg Larsen. Personal informatics in the wild: Hacking habits for health & happiness. In *Proceedings of Extended Abstracts on Human Factors in Computing Systems*, pages 3179–3182. ACM, 2013.

[LGHL10] Sangkeun Lee, Gihyun Gong, Inbeom Hwang, and Sang-goo Lee. Lifelogon: A practical lifelog system for building and exploiting lifelog ontology. In *Proceedings of the International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing*, pages 367–373. IEEE, 2010.

[LHD$^+$11] Ian Li, Kristina Höök, Anind Dey, Yevgeniy Medynskiy, and Jodi Forlizzi. Personal informatics and hci: Design, theory, and social implications. In *Proceedings of Extended Abstracts on Human Factors in Computing Systems*, pages 2417–2420. ACM, 2011.

[LMFL12] Ian Li, Yevgeniy Medynskiy, Jon Froehlich, and Jakob Eg Larsen. Personal informatics in practice: Improving quality of life through data. In *Proceedings of Extended Abstracts on Human Factors in Computing Systems*, pages 2799–2802. ACM, 2012.

[LML$^+$06] James J. Lin, Lena Mamykina, Silvia Lindtner, Gregory Delajoux, and Henry B. Strub. Fish'n'steps: Encouraging physical activity with an interactive computer game. In *Proceedings of the 8th international conference on Ubiquitous Computing*, pages 261–278. Springer-Verlag, 2006.

[Man97] Steve Mann. An historical account of the "wearcomp" and "wearcam" projects developed for "personal imaging". In *Proceedings of the 1st IEEE International Symposium on Wearable Computers*, page 66. IEEE, 1997.

[Nie12] Jakob Nielsen. Thinking aloud: The #1 usability tool, 2012. http://www.nngroup.com/articles/thinking-aloud-the-1-usability-tool.

[OHRM09] Erica L. Olmsted-Hawala, Jennifer C. Romano, and Elizabeth D. Murphy. The use of paper-prototyping in a low-fidelity usability study. In *Proceedings of IEEE International Professional Communication Conference*, pages 1–11. IEEE, 2009.

[PHS13] Afarin Pirzadeh, Li He, and Erik Stolterman. Personal informatics and reflection: A critical examination of the nature of reflection. In *Proceedings of Extended Abstracts on Human Factors in Computing Systems*, pages 1979–1988. ACM, 2013.

[PTH13] Dana Pavel, Dirk Trossen, and Matthias Holweg. Lifestyle stories: Correlating user information through a story-inspired paradigm. In *Proceedings of the 7th International Conference on Pervasive Computing Technologies for Healthcare*, pages 412–415. ICST, 2013.

[PV97] James O. Prochaska and Wayne F. Velicer. The transtheoretical model of health behavior change. *American Journal of Health Promotion*, 1997.

[RLJ$^+$11] M. S. Ryoo, Jae-Yeong Lee, Ji Hoon Joung, Sunglok Choi, and Wonpil Yu. Personal driving diary: Constructing a video archive of everyday driving events. In *Proceedings of IEEE Workshop on Applications of Computer Vision (WACV)*, pages 628–633, 2011.

[SHKK10] Inchul Song, Deokmin Haam, Hangkyu Kim, and Myoung Ho Kim. Ontlms: An ontology-based lifelog management system. In *Proceedings of the 12th International Asia-Pacific Web Conference*, pages 341–343. IEEE, 2010.

[SHM10] Esmadi Abu Abu Seman, Idyawati Hussein, and Murni Mahmud. Why thinking aloud matters for usability evaluation? In *Proceedings of the 2nd International Conference on Computer Engineering and Applications (ICCEA)*, pages 462–466. IEEE, 2010.

[SKS+13] Yusuke Satonaka, Takumi Kitazawa, Kazuki Suzuki, Yuki Fukizaki, Takuya Azumi, and Nobuhiko Nishio. Lifelog-based active movement assistant system. In *Proceedings of the 1st International Conference on Cyber-Physical Systems, Networks, and Applications (CPSNA)*, pages 7–12. IEEE, 2013.

[SM10] Brian Still and John Morris. The blank-page technique: Reinvigorating paper prototyping in usability testing. In *Proceedings of IEEE Transactions of Professional Communication*, pages 144–157. IEEE, 2010.

[SSS+14] Arkadiusz Stopczynski, Vedran Sekara, Piotr Sapiezynski, Andrea Cuttone, Mette My Madsen, Jakob Eg Larsen, and Sune Lehmann. Measuring large-scale social networks with high resolution. *PLOS ONE [DOI: 10.1371/journal.pone.0095978]*, 2014.

[SW10] Abigail Sellen and Steve Whittaker. Beyond total capture: A constructive critique of lifelogging. *Communications of the ACM*, pages 70–77, 2010.

[SZC12] Schwanda Victoria Sosik, Xuan Zhao, and Dan Cosley. See friendship, sort of: How conversation and digital traces might support reflection on friendships. In *Proceedings of the ACM 2012 conference on Computer Supported Cooperative Work*, pages 1145–1154. ACM, 2012.

[TM13] Go Tanaka and Hiroshi Mineno. A method of estimating outdoor situation for lifelog generation. *Proceedings of the 2nd Global Conference on Consumer Electronics (GCCE)*, pages 361–362, 2013.

[TMA+08] Katsuhiro Takata, Jianhua Ma, Bernady O. Apduhan, Runhe Huang, and Norio Shiratori. Lifelog image analysis based on activity situation models using contexts from wearable multi sensors. In *Proceedings of the International Conference on Multimedia and Ubiquitous Engineering*, pages 160–163. IEEE, 2008.

[Tra12] Brian Traynor. Rapid paper prototyping: 100 design sketches in 10 minutes, 18 designs presented, 6 prototypes tested, student engagement – priceless! In *Proceedings of the IEEE International Professional Communication Conference*, pages 1–5. IEEE, 2012.

[TSMN12] Kohei Takahashi, Akira Shimojo, Shinsuke Matsumoto, and Masahide Nakamura. Mashmap: Application framework for map-based visualization of lifelog with location. In *Proceedings of the 9th Asia-Pacific Symposium on Information and Telecommunication Technologies (APSITT)*, pages 1–6. IEEE, 2012.

[WKP$^+$12] Steve Whittaker, Vaiva Kalnikaitė, Daniella Petrelli, Abigail Sellen, Nicolas Villar, Ofer Bergman, Paul Clough, and Jens Brockmeier. Socio-technical lifelogging: Deriving design principles for a future proof digital past. *Human-computer Interaction [DOI: 10.1080/07370024.2012.656071]*, 2012.

[YI09] Kiichiro Yamano and Katunobu Itou. Browsing audio life-log data using acoustic and location information. In *Proceedings of the 3rd International Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies*, pages 96–101. IEEE, 2009.

[ZZXM09] Yu Zheng, Lizhu Zhang, Xing Xie, and Wei-Ying Ma. Mining interesting locations and travel sequences from gps trajectories. In *Proceedings of the 18th international conference on World wide web*, pages 791–800. ACM, 2009.