Chair for Computer Science 10 (Media Computing and Human-Computer Interaction)

Prototyping an On-Demand Routing App with Location-Based Content Push

Bachelor's Thesis
submitted to the
Media Computing Group
Prof. Dr. Jan Borchers
Computer Science Department
RWTH Aachen University

by
Sven Titgemeyer

Thesis advisor:
Prof. Dr. Jan Borchers

Second examiner:
Prof. Dr. Schuh

Registration date: 19.08.2018
Submission date: 10.01.2019

Eidesstattliche Versicherung

Name, Vorname	Matrikelnummer

Ich versichere hiermit an Eides Statt, dass ich die vorliegende Arbeit/Bachelorarbeit/ Masterarbeit* mit dem Titel

selbständig und ohne unzulässige fremde Hilfe erbracht habe. Ich habe keine anderen als die angegebenen Quellen und Hilfsmittel benutzt. Für den Fall, dass die Arbeit zusätzlich auf einem Datenträger eingereicht wird, erkläre ich, dass die schriftliche und die elektronische Form vollständig übereinstimmen. Die Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Ort, Datum	Unterschrift

*Nichtzutreffendes bitte streichen

Belehrung:

§ 156 StGB: Falsche Versicherung an Eides Statt

Wer vor einer zur Abnahme einer Versicherung an Eides Statt zuständigen Behörde eine solche Versicherung falsch abgibt oder unter Berufung auf eine solche Versicherung falsch aussagt, wird mit Freiheitsstrafe bis zu drei Jahren oder mit Geldstrafe bestraft.

§ 161 StGB: Fahrlässiger Falscheid; fahrlässige falsche Versicherung an Eides Statt

(1) Wenn eine der in den §§ 154 bis 156 bezeichneten Handlungen aus Fahrlässigkeit begangen worden ist, so tritt Freiheitsstrafe bis zu einem Jahr oder Geldstrafe ein.

(2) Straflosigkeit tritt ein, wenn der Täter die falsche Angabe rechtzeitig berichtigt. Die Vorschriften des § 158 Abs. 2 und 3 gelten entsprechend.

Die vorstehende Belehrung habe ich zur Kenntnis genommen:

Ort, Datum	Unterschrift

Contents

List of Figures

Abstract

Previous work showed that there is a need for multi-stop route planning, especially (but not only) in a tourism context. Recent progress in technology allows us to build more dynamic applications that can use advanced touch technology to manipulate routes and provide more context-aware dynamic information by using fast mobile internet access, the users location and their preferences.

In this thesis we perform an analysis of related apps in the field and iteratively develop a working prototype of a multi-stop route planning app.

We found that providing the user with more information helps the user to build a route that fits into its schedule. We identified many common patterns in existing route planning apps with which users are already familiarized. Static content can be well mixed with dynamic, often time-sensitive, content and offers.

Überblick

Vorhergehende Arbeit hat gezeigt, dass es einen Bedarf für Routenplanung mit mehreren Stationen gibt, insbesondere (aber nicht ausschließlich) in einem touristischen Kontext. Durch den Fortschritt der Technologie ist es möglich, dynamischere Anwendungen zu entwickeln, welche fortschrittliche Touch-Technologie zur Bearbeitung von Routen nutzen. Mit Hilfe von schnellem mobilen Internet, des Standorts des Users und seiner Interessen ist es möglich mehr kontextbezogene dynamische Informationen bereitzustellen.

In dieser Arbeit untersuchen wir verwandte Apps aus der Domäne und entwickeln iterativ einen funktionalen Prototypen für eine Routenplanungsapp mit mehreren Stopps.

Wir haben herausgefunden, dass es für den Nutzer hilfreich ist, wenn er mehr Informationen zu seiner Route erhält. Wir konnten Gemeinsamkeiten zwischen den existierenden Apps finden, welche wir nutzen können, damit der Nutzer sich direkt in der App zurecht findet. Der statische Inhalt kann gut mit dynamischen, oft zeit-kritischen, Inhalten und Angeboten verbunden werden.

Acknowledgements

Thank you to Philipp Wacker and Sebastian Hueber, my supervisors, for the feedback throughout the thesis.

Thank you to everyone at e.Go Digital. It was super fun to work with you.

Special thanks to everyone who participated in the user study and provided feedback to the countless number of prototypes.

Thanks to Prof. Dr. Jan Borchers and Prof. Dr. Günther Schuh for examining this thesis.

And finally, very special thanks to Van Dyck who roasted the coffee beans without which this thesis wouldn't have been possible.

Conventions

Throughout this thesis we use the following conventions.

Text conventions

Definitions of technical terms or short excursus are set off in coloured boxes.

EXCURSUS:
Excursus are detailed discussions of a particular point in a book, usually in an appendix, or digressions in a written text.

Definition: *Excursus*

Source code and implementation symbols are written in typewriter-style text.

```
myClass
```

The whole thesis is written in American English. We use the plural form for the first person. For unidentified third persons we use the pronoun they/their.

Chapter 1

Introduction

Many use-cases that require going to multiple locations can benefit from better route planning. These locations can be sights, virtual bus stops or appointments on your calendar. Usually, you have a time span available and a number of things you want to do within that time span. In a tourist context, this might mean you have one day in a city and you want to make the most of it. You need to decide what stations you want to visit and in which order and you want to be able to change your decision at any time.

1.1 Open Questions

We want to analyze what information a user needs in order to plan a trip that involves visiting multiple stations. Most navigation apps do only provide information on how long travel between stations takes. Therefore we ask the question, if a routing app that allows the user to plan an entire journey, including travel times and expected stay durations, can help the user to plan a trip.

Previous research indicated that tourism apps benefit from dynamic, context-aware content. We want to analyze, how tourism context and offers can be presented to the user in a meaningful way.

1.2 Outline

In this thesis, we analyze apps related to route planning and tourism. We find common interaction patterns and use the knowledge the user already has about navigation apps.

We build a route-planning prototype using a DIA process. We start on simple paper prototypes and evolve them into a functional app. We verify our assumptions on how users want to plan a route using a user study. Finally, we look into how we can integrate location-based content.

Chapter 2

Related work

There has been a lot of research on planning routes. Often, in the context of tourist guides which are used to plan trips that usually include a lot of stations. One of the earliest projects we looked into here is the GUIDE project by Lancaster University [Cheverst et al., 2000].

The GUIDE project researched requirements for a tourist guide.

The GUIDE project ran on `TeamPad` and provided information related to common tourist attractions. They performed an extensive requirements study and identified four key requirements for a context-aware tourist guide:

- Flexibility

  They found that a tourist guide must be flexible enough to adapt to the usage patterns of the User. E.g. some users prefer complete guided tours, others want the flexibility to create completely new tours.

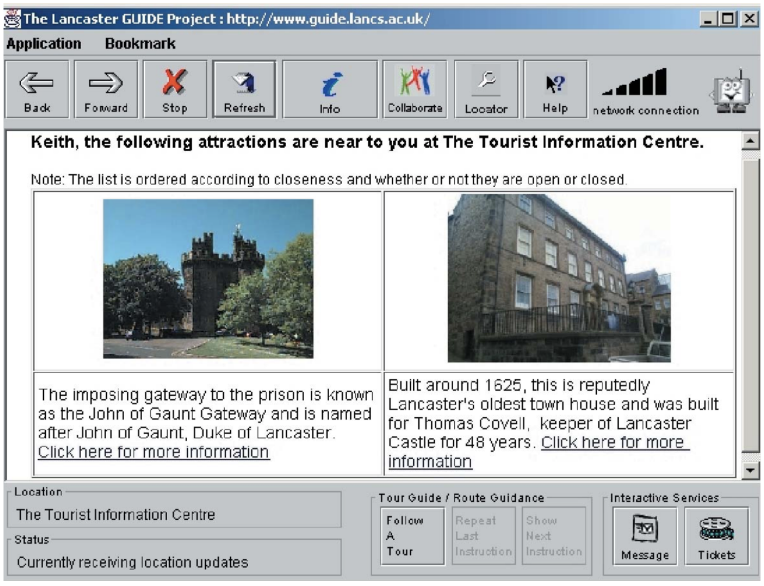
- Context-Aware Information

  A tourist guide must take personal and environmental aspects, like the users interests or opening times of attractions, into account.

- Support For Dynamic Information

  Location-Based offers are a key requirement.

  Their research found, that a tourist guide must show things like daily specials at a citys café. That's one of the things we call "Location-Based Content Push".



Keith, the following attractions are near to you at The Tourist Information Centre.

Note: The list is ordered according to closeness and whether or not they are open or closed.

The imposing gateway to the prison is known as the John of Gaunt Gateway and is named after John of Gaunt, Duke of Lancaster. Click here for more information

Built around 1625, this is reputedly Lancaster's oldest town house and was built for Thomas Covell, keeper of Lancaster Castle for 48 years. Click here for more information

The Tourist Information Centre

Currently receiving location updates

Figure 2.1: The GUIDE user interface showing the dynamically created Nearby Attractions page [Cheverst et al., 2002].

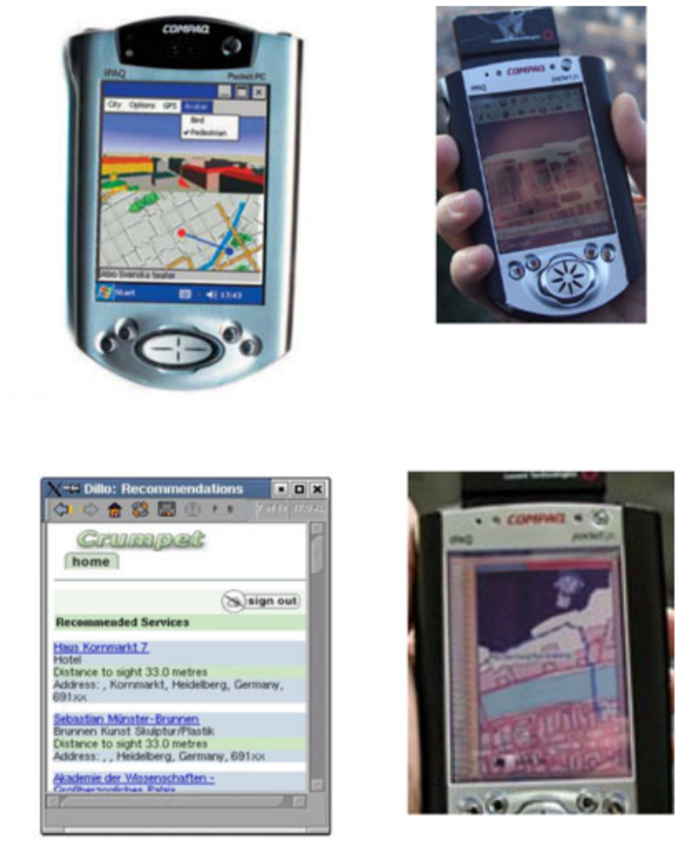
- Support for Interactive Services

  They observed that tourists commonly go back to the tourist info, for example to make bookings for accomodation or travel.

Modern technology is better suited to implement these requirements.

Looking at these requirements we found that the advance of technology in the past decades should allow to implement these requirements in a mobile app. We have fast mobile data, GPS, and lots of APIs to available data sources and booking portals that haven't been available at the time of the GUIDE project. Tourists are willing to spend money for the interactive services and are interested in offers around their location which also provides monetarization possibilities for a good tourist guide.

There have been many tourist guides since the GUIDE project which can broadly be categorized into generations depending on the technology they had available.



Recommended Services

Haus Kornmarkt 7
Hotel
Distance to sight 33.0 metres
Address: , Kornmarkt, Heidelberg, Germany, 691xx

Sebastian Münster-Brunnen
Brunnen Kunst Skulptur/Plastik
Distance to sight 33.0 metres
Address: , , Heidelberg, Germany, 691xx

Akademie der Wissenschaften -

Figure 2.2: Tourist guides for PDAs suffered from the poor technology in contrast to today [Kenteris et al., 2011].

PDAs were limited by their screen size and capabilities.

Looking at tourist guides for PDAs (see Figure 2.2) we see that they might have provided good information, but their user interfaces are hard to use. The screen sizes and resolutions were to small, the touchscreens weren't as good as today. More on the evolution of tourist guides (before Smartphones) can be found in [Kenteris et al., 2011].

2.1 Algorithmic Tourist Guides

Algorithmic tourist guides create routes automatically.

There has also been research on building tourist routes algorithmically, sometimes referred to as the Tourist Trip Design Problem (TTDP) [Gavalas et al., 2014]. These approaches feed information about the user and the constraints (like time available, location etc.) into an algorithm and try to build a trip tailored to the user on-demand. Although we are not looking into ways to build tourist trips algorithmically here, we think these kind of automatic systems would need an interface which allow the user to modify the generated routes.

There is a need for interfaces to modify those routes.

Schaller looked into how the user can maintain control over the generated route by providing him with an interface to modify the route [Schaller, 2014]. In their interface the user is able to modify the time they wishes to stay at a station and to change the order of stations (see 2.3).

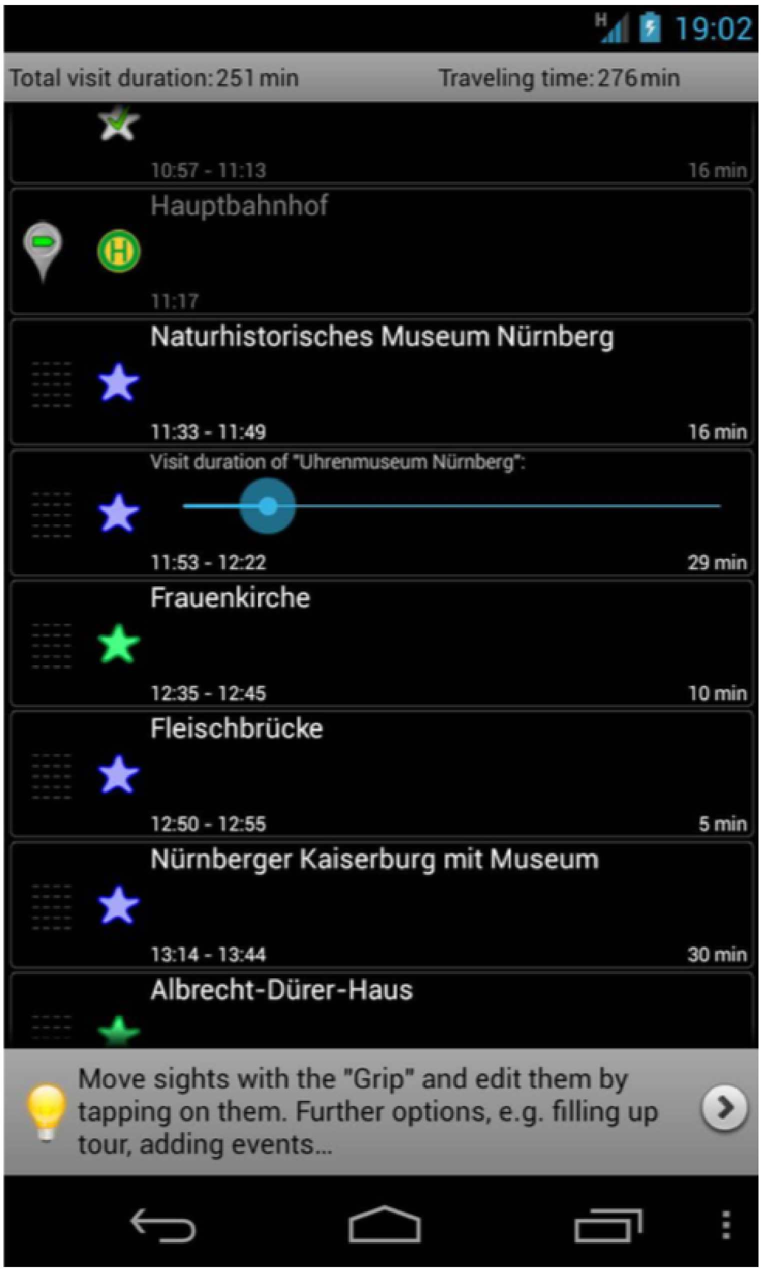

19:02
Total visit duration: 251 min
Traveling time: 276 min
10:57 - 11:13
16 min
Hauptbahnhof
11:17
Naturhistorisches Museum Nürnberg
11:33 - 11:49
16 min
Visit duration of "Uhrenmuseum Nürnberg":
11:53 - 12:22
29 min
Frauenkirche
12:35 - 12:45
10 min
Fleischbrücke
12:50 - 12:55
5 min
Nürnberger Kaiserburg mit Museum
13:14 - 13:44
30 min
Albrecht-Dürer-Haus
Move sights with the "Grip" and edit them by tapping on them. Further options, e.g. filling up tour, adding events...

Figure 2.3: Schaller built an app that could algorithmically create a route based on a few user provided constraints and allows the user to modify that route. Users can set the duration they want to stay at a station and the color tells them how close they are to the recommended durations [Schaller, 2014].

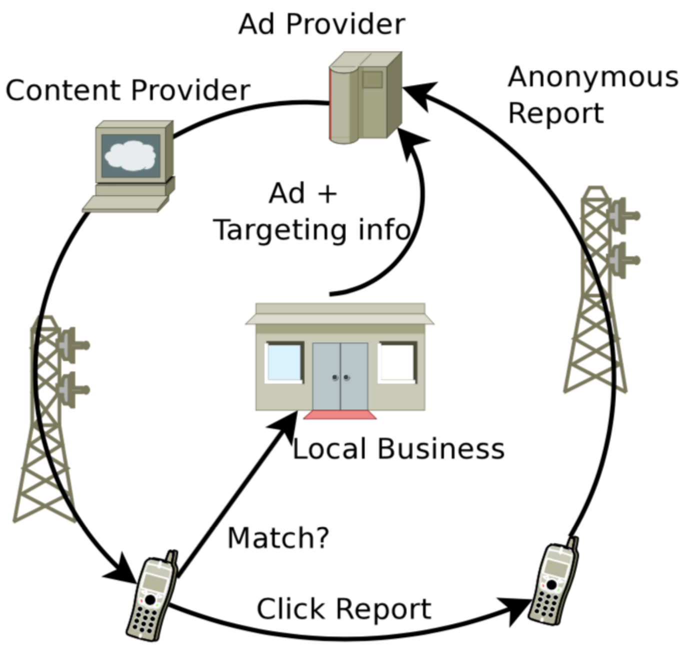


Figure 2.4: Location-based content push might not only be a way to monetize an app, but eventually be exactly what the user expects from such an app. Haddadi et al. [2010] looked into how such a service can monetize offers around the user while still maintaining their privacy [Haddadi et al., 2010].

2.2 Location-Based Content Push

Offers the user wants are not perceived as Ads.

As the GUIDE project already found out, people using a tourist guide - or planning a trip in general - are often looking for additional content around their current location or route. This content could also include offers, which are monetized but still actively wanted by the user, thus not perceived as “ads”.

Location data can be used to show offers without compromising privacy.

Haddadi et al. [2010] analyzed how location-aware data can be used to show the user relevant information, e.g. relating to local businesses, in a privacy sensitive way (see Figure 2.4). Personalized offers along the route might be exactly what the user wants, but it might also be perceived as an invasion into the users privacy, especially if the offer shown is irrelevant to the user. O'Donnell analyzed the perception of personalized apps in detail [O'Donnell and Cramer, 2015].

Chapter 3

Analysis of Related Apps

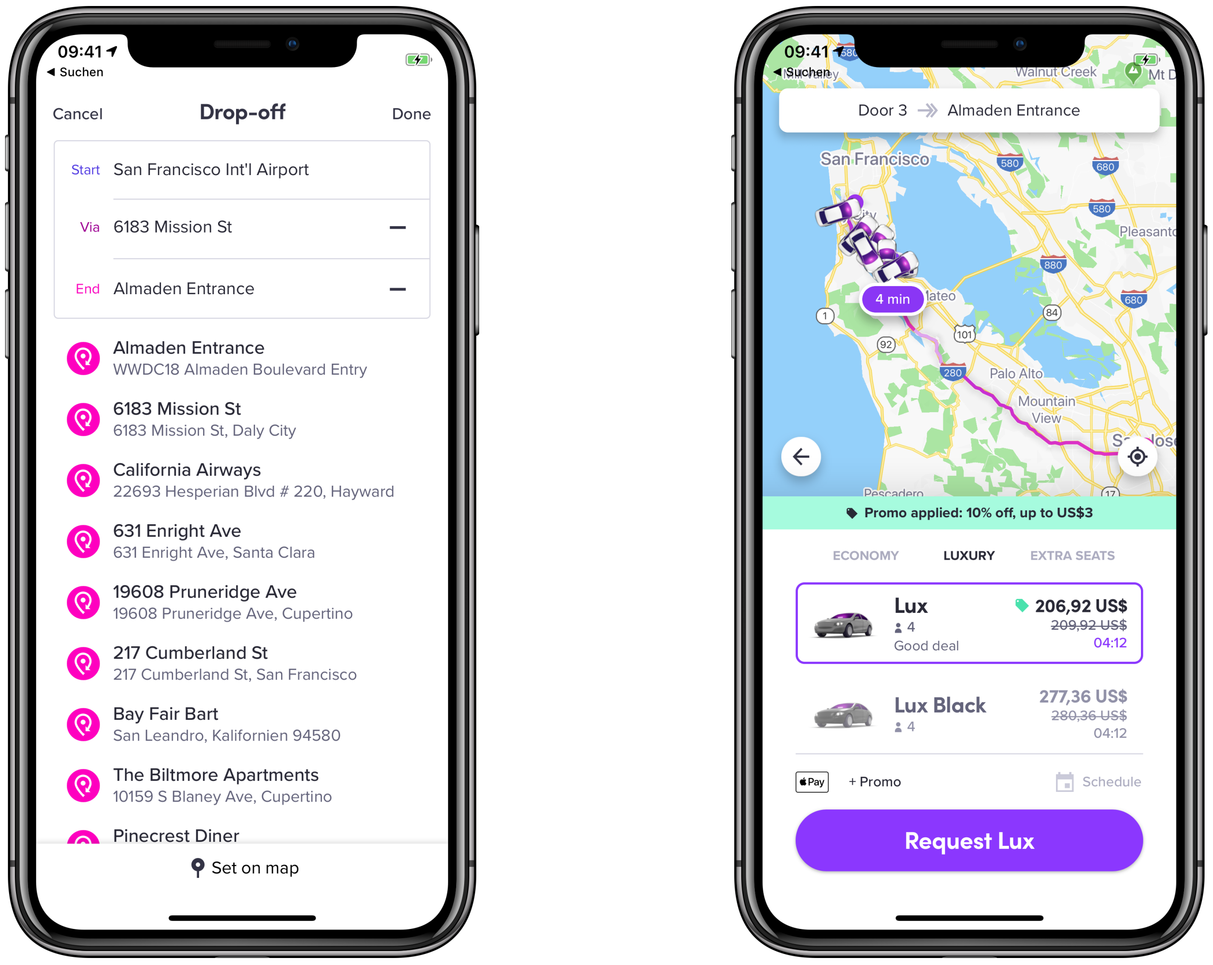
We started our work by researching other apps, that require route planning. We tried to find common interaction patterns which work well. The apps we looked into can be broadly categorized into two categories: Apps that provide transportation between stations and apps that provide content for stations.

3.1 Lyft

Lyft is an on-demand transportation app providing over 1 million rides per day [Wik]. Its main purpose is to provide rides from one station to another, although it is possible to add one additional stop. There is no additional content related to the stations provided.

Lyft uses a bottom sheet, a map and user can have three stations.

Most of the screen space is covered by a map, roughly the bottom third is used to search for stations (see 3.1). When the user selects the search field the bottom sheet transitions to fullscreen. It is possible to select a start and destination stations and there is a button for adding one additional stop. It is not possible to see the map and selected stations at the same time.


09:41
◀ Suchen

Cancel **Drop-off** Done

Start San Francisco Int'l Airport
Via 6183 Mission St —
End Almaden Entrance —

Almaden Entrance
WWDC18 Almaden Boulevard Entry

6183 Mission St
6183 Mission St, Daly City

California Airways
22693 Hesperian Blvd # 220, Hayward

631 Enright Ave
631 Enright Ave, Santa Clara

19608 Pruneridge Ave
19608 Pruneridge Ave, Cupertino

217 Cumberland St
217 Cumberland St, San Francisco

Bay Fair Bart
San Leandro, Kalifornien 94580

The Biltmore Apartments
10159 S Blaney Ave, Cupertino

Pinecrest Diner

Set on map

09:41
◀ Suchen

Door 3 → Almaden Entrance

4 min

Promo applied: 10% off, up to US$3

ECONOMY LUXURY EXTRA SEATS

Lux 4 Good deal — 206,92 US$ ~~209,92 US$~~ 04:12

Lux Black 4 — 277,36 US$ ~~280,36 US$~~ 04:12

Pay + Promo Schedule

Request Lux

Figure 3.1: The route is shown fullscreen, one additional station can be added.

3.2 Uber

Uber was founded 2009 and provides on-demand transportation. Users are able to select a start and destination and book a ride. Uber works with private drivers that will drive you to your destination.

Big map and bottom sheet.

When you open the app you see a big map and a route planning UI at the top. It is possible to add up to 3 stops, but you don't get any information on when you will arrive at a stop.

Add stations using Search or Map.

Stations can be added using a search functionality or by adding them on the map. If you want to add them using the map you have to open the search and tap "Select on map" at the bottom. You get a fixed pin and can move the map under the pin. You get live reverse geocoding when you move the map and can confirm once you found your pickup point.

After you created a route you are presented a bottom sheet in which you can select which kind of car you want to book. There's a big booking button on the bottom of the sheet.

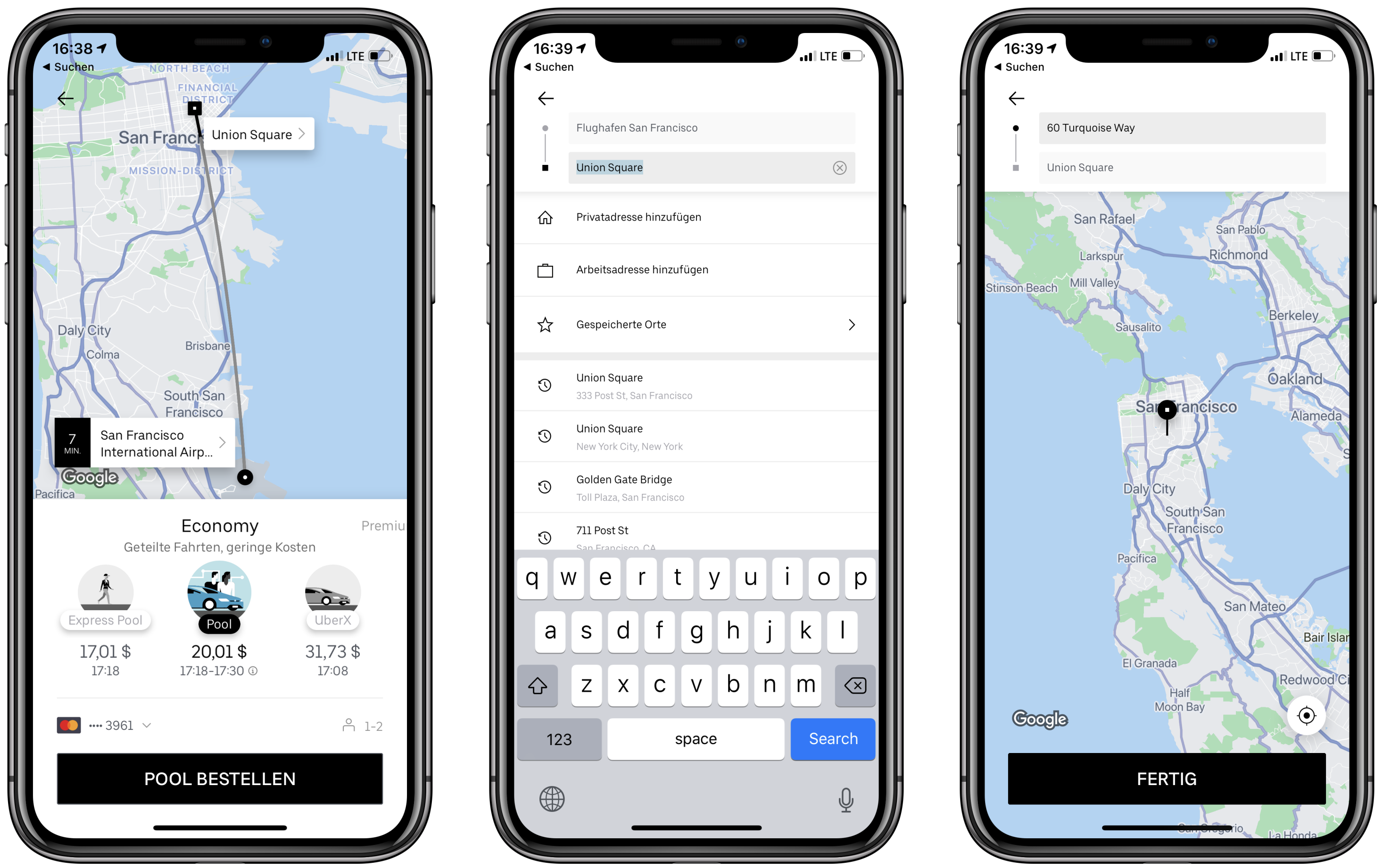


Figure 3.2: The UI switches between map and route planning.

3.3 DB Navigator

DB Navigator is the app of German railway-company, which is the second-largest transport company in the world [db]. DB Navigator is especially relevant in the context of multi-stop route planning, because due to the nature of how trains work you always need to plan multiple stations and arrival times / connection times.

Screen divided into map and planning.

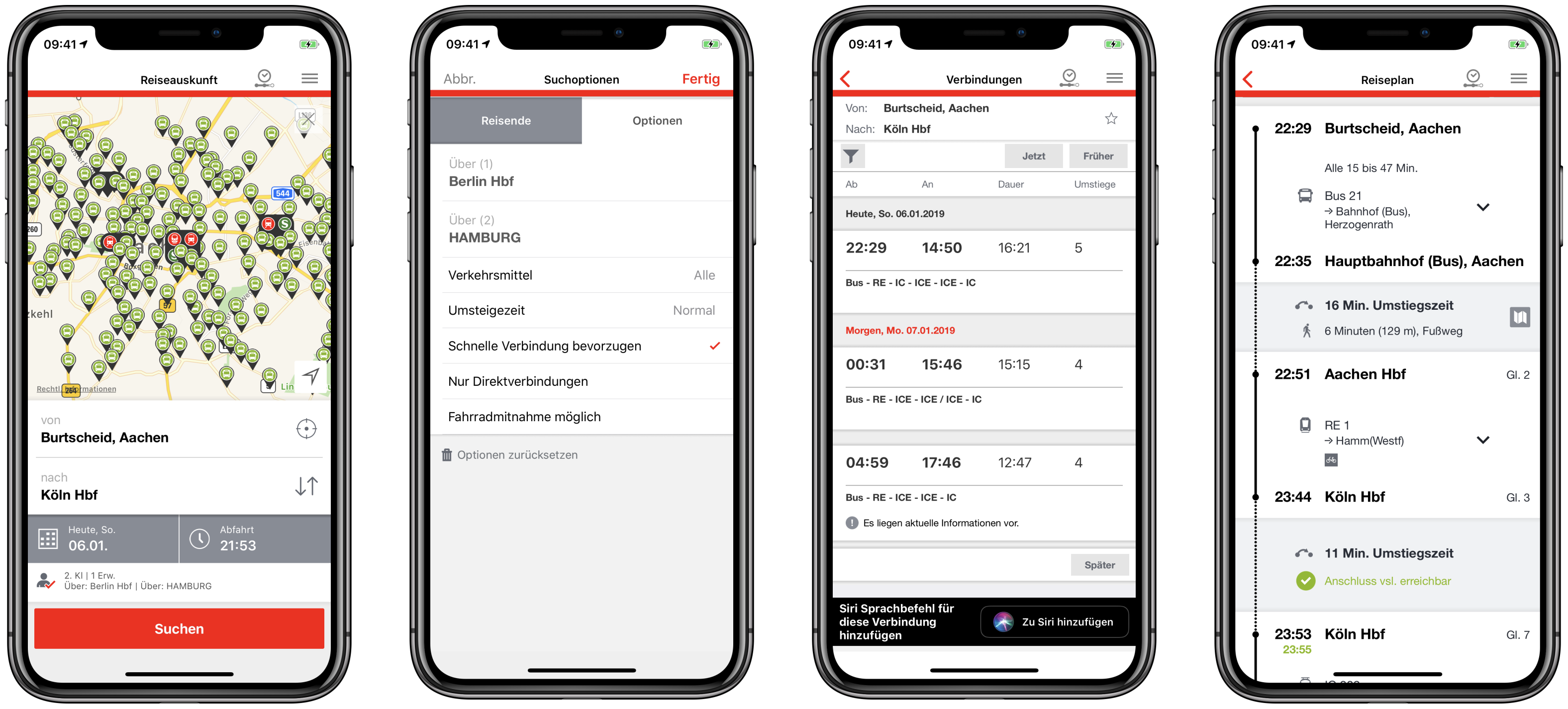
When you start the app, you can select a start and end destination, the screen is divided into a map and a planning UI at the bottom. It is possible to add additional stops, but it's a rather complicated multi-step process using the menu.

Connections on fixed schedule.

Once you have select your stops you get shown a list of possible connections. In contrast to on-demand apps, these connections are build based on fixed schedules. In the list of connections, you can see start and arrival time as well as the duration of the trip.

There's very detailled information to every available route.

When you tap on one of the available connections you get detailed informations. They show you every stop that is on your route, less important stops are minimized and can be expanded. For every step they show you arrival times. When you have to switch trains, they show you the time between trains. These routes are fixed and can not be modified by the user.


09:41

Reiseauskunft

von
Burtscheid, Aachen

nach
Köln Hbf

Heute, So.
06.01.

Abfahrt
21:53

2. Kl | 1 Erw.
Über: Berlin Hbf | Über: HAMBURG

Suchen

09:41

Abbr. Suchoptionen Fertig

Reisende Optionen

Über (1)
Berlin Hbf

Über (2)
HAMBURG

Verkehrsmittel	Alle
Umsteigezeit	Normal
Schnelle Verbindung bevorzugen	✓
Nur Direktverbindungen	
Fahrradmitnahme möglich	

Optionen zurücksetzen

09:41

Verbindungen

Von: Burtscheid, Aachen
Nach: Köln Hbf

Jetzt Früher

Ab	An	Dauer	Umstiege
Heute, So. 06.01.2019			
22:29	14:50	16:21	5
Bus - RE - IC - ICE - ICE - IC			
Morgen, Mo. 07.01.2019			
00:31	15:46	15:15	4
Bus - RE - ICE - ICE / ICE - IC			
04:59	17:46	12:47	4
Bus - RE - ICE - ICE - IC			

Es liegen aktuelle Informationen vor.

Später

Siri Sprachbefehl für diese Verbindung hinzufügen

Zu Siri hinzufügen

09:41

Reiseplan

22:29 Burtscheid, Aachen

Alle 15 bis 47 Min.

Bus 21
→ Bahnhof (Bus), Herzogenrath

22:35 Hauptbahnhof (Bus), Aachen

16 Min. Umstiegszeit
6 Minuten (129 m), Fußweg

22:51 Aachen Hbf Gl. 2

RE 1
→ Hamm(Westf)

23:44 Köln Hbf Gl. 3

11 Min. Umstiegszeit
Anschluss vsl. erreichbar

23:53 Köln Hbf Gl. 7
23:55

Figure 3.3: In DB Navigator users can add multiple stations and select from predefined routes.

3.4 Komoot

Komoot is a bicycle and hiking app.

Komoot is an app to plan hiking or bicycle routes. According to the founder, the main audience are people living in big cities who want to cycling or hiking tours but don't know where to go [Schnor]. We looked into this app because it combines multi-stop route planning and exploring locations.

It features a big map with annotations.

On app start you see a big map with interesting locations and prepared routes marked on the map. Stations are annotated with a point annotation, prepared routes show either a bicycle or a hiker in the annotation view.

Information related to the stations is shown in a bottom sheet.

On tapping an annotation, you get presented a bottom sheet describing the POI and can add is as start or destination to your route. After adding the station to your route a planning UI opens in fullscreen. It features general settings (like cycling or hiking) and a list of the selected stations. The next station is displayed as a search field. When you select it, you get a fullscreen search, where you can either search for your next station or choose "Select on map".

If you choose to select a station on the map you get a similar interface to the one at the beginning where you get information for each POI in a bottom sheet.

Additional information shown for the total route.

When you have created a route the planning UI shows you all your stations at the top and additional information like total duration, distance, elevation on the bottom.

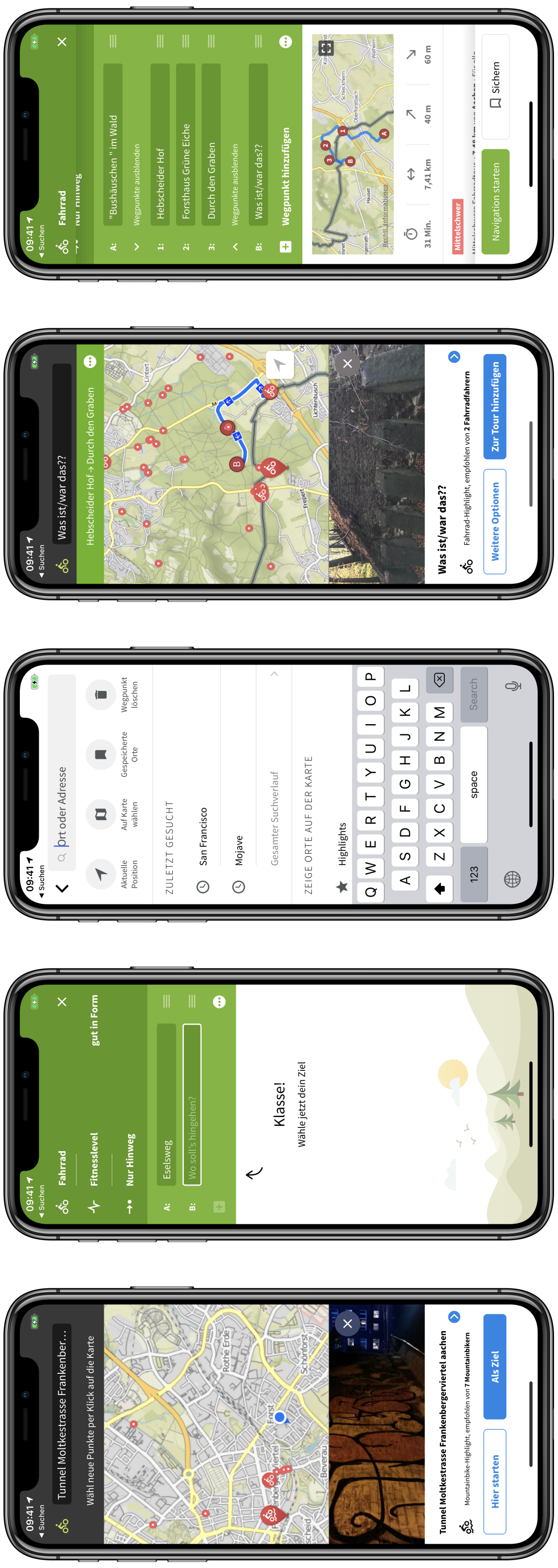


Figure 3.4: In Kommot users can create bicycle or hiking routes. It features predefined routes and individual POIs.

3.5 Apple Maps

Apple Maps is the preinstalled navigation app on iOS. We assume most users to be familiar with its UI and want the user to be able to apply their knowledge of Apple Maps when using our app.

Shows a map and bottom sheet with search.

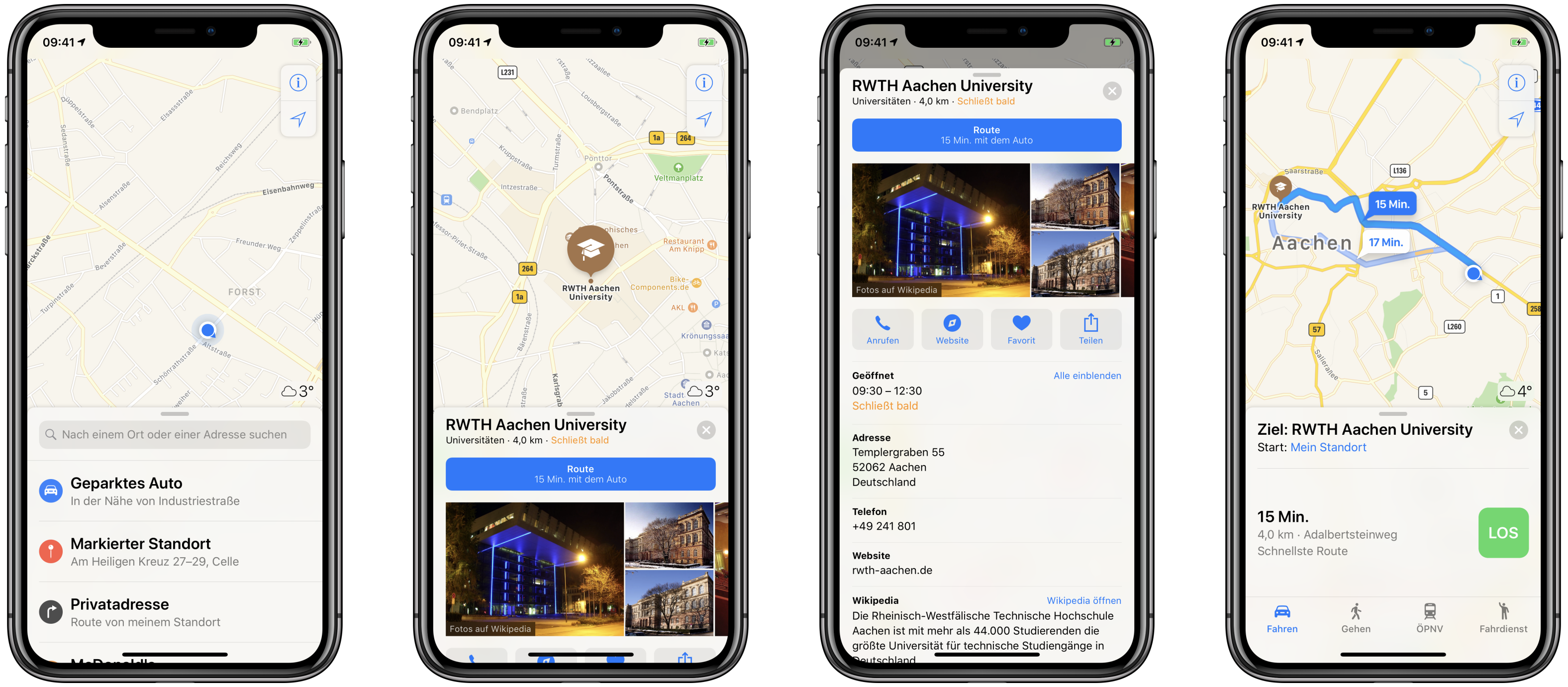
Apple Maps features a bottom sheet with a search bar on top. Users can either select stations on the map or search for stations using the search. Siri suggested locations and recent stations are already shown before entering a search text.

POIs are presented in bottom sheets.

When a POI is selected the user is presented with an additional bottom sheet that shows information related to the POI and a button to start a navigation to that station.

There are no additional stops fore routes.

It's not possible to add multiple stops to a route, therefore the route planning interface doesn't consist of more than a start and destination station and total duration.



Geöffnet
09:30 – 12:30
Schließt bald

Adresse
Templergraben 55
52062 Aachen
Deutschland

Telefon
+49 241 801

Website
rwth-aachen.de

Wikipedia
Die Rheinisch-Westfälische Technische Hochschule Aachen ist mit mehr als 44.000 Studierenden die größte Universität für technische Studiengänge in Deutschland

15 Min.
4,0 km · Adalbertsteinweg
Schnellste Route

Figure 3.5: Apple Maps shows a big map and all other content is placed in bottom sheets. The user can resize the sheet from very small to nearly fullscreen.

3.6 Google Maps

Google Maps is an app developed by Google that can be used to explore POIs in a city and for navigation. It features a big map, a search bar on top and a bottom sheet on the bottom. On app start, it offers general information around the current location (or city).

Google Maps also uses bottom sheets.

When the user selects a POI it shows information in a bottom sheet and an option to add that station to the current route.

Multi stop routes, but limited information.

A route can consist of multiple stations which are shown in an overlay at the top. When multiple stations are added, it shows the total duration of the trip, but no arrival times at individual stations.

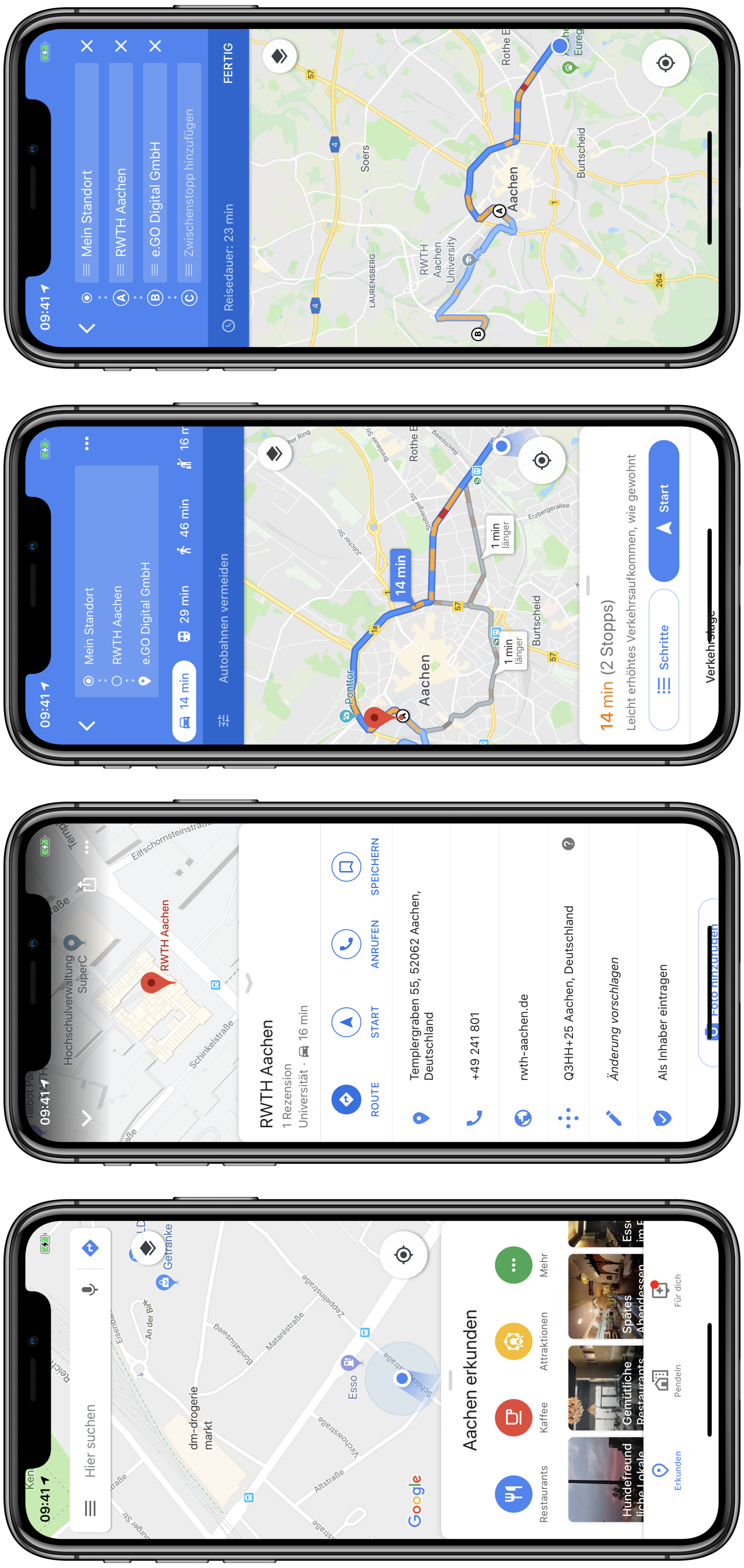


Figure 3.6: Google Maps features a big map, a bottom sheet and multi-stop route planning.

3.7 Discussion of Related Apps

After an in-depth analysis of all the related apps in the field, we isolated a few key aspects for developing our app. We want to mention the most important aspects here. Of course, there are more similarities, between the apps analyzed and the one we build, but we want to highlight the key learnings.

3.7.1 Map

All apps show content on a Map.

All apps show each of the selected stations on a map. The route between the stations is drawn as an overlay on the map. While most apps show a specific, calculated route, Uber only shows an arc from start to destination.

3.7.2 Connection between Route and Map

There's always some other representation besides the map.

All apps that include route planning show the route on the map as well as in some other, more textual presentation which is usually also be used for editing the route. Some Apps allow working with the textual representation while seeing the map at the same time (In contrast to others that always transition to fullscreen, obstructing the map) which is a big benefit for usability, especially when working with multiple stations.

Definition: *Planning*

PLANNING:
From now on, we will refer to the textual representation of the planning UI as "Planning". Similarly, we will refer to the map as "Map".

All apps use bottom sheets.

Many apps use a bottom sheet, which covers about the bottom third of the screen (on an iPhone X) and sometimes can be moved. That way, the user can decide themself if they wants to see more of the map or more of the planning.

BOTTOM SHEET:
A bottom sheet is a UI Element, that shows an additional view controller on top of a main view controller, floating from the bottom. An Apple example can be seen in the iOS 12 Maps, Stocks or Shortcuts app.

Definition:
Bottom Sheet

Chapter 4

Building the Route Planning app

There are different use-cases and different users for a route planning app. Some users might want to plan an exact trip through a city. Other users might just want to explore what a city has to offer while other users might not want to plan anything at all and just want to know how to get to a single destination.

Different users have different use-cases we want to cover.

4.1 Requirements

In order to provide the user with all the information they needs to fulfill its task, we found these requirements:

Users must be able to add and remove stations to the route. They need to understand how and why a station has been added to a route. They might change their mind while planning a trip, therefore it must be easy to remove stations or to reorder them.

Add and Remove stations.

Users must know the travel time from station to station. The travel time can be different based on the mode of transportation they choose.

Travel Times must be shown.

Users need to be able to plan travel times and times they stay at stations/

Especially with multiple stations users must be able to calculate the time they stay at stations. Many route planning apps do only calculate travel times and add them together. However, the user will usually stay at a station for a while, then decide to move on to the next station. Without a way to specify how long they plan to stay there, they have to add up many numbers themselves. It is crucial, to provide users with information how long people usually spend at those stations.

4.2 DIA Process

We used DIA Cycles to develop the app.

We developed the app using a DIA process. In each cycle, we designed a feature, implemented it (either in Software or on paper) and analyzed what we did. We used the results of our analysis in the following cycle to validate that our product is going into the right direction. We tried to keep our cycles as short as possible, especially in the early cycles. Therefore, we started by doing our prototypes on paper. Because drawing on paper is extremely fast, the first cycles lasted often just a few minutes and provided us with valuable feedback which dramatically shaped our later, more costly, prototypes.

4.3 Paper Prototypes

We used a map and bottom sheet in our first paper prototypes.

For our first prototypes, we decided we need a map to show the route and Points-of-Interest in the background. The route is also shown in a bottom sheet. The stations can be reordered similar to the standard `UITableView` reordering and there is a search field after the last station. Initially, we showed the information relating to stations in a Callout on the map (see Figure 4.1).

We need to show more information to make sure the user understands their time constraints.

We quickly realized, that the user needs much more information in order to understand the time constraints on their route. In order to fully understand the time constraints on their route, they need to know the start time, duration be-

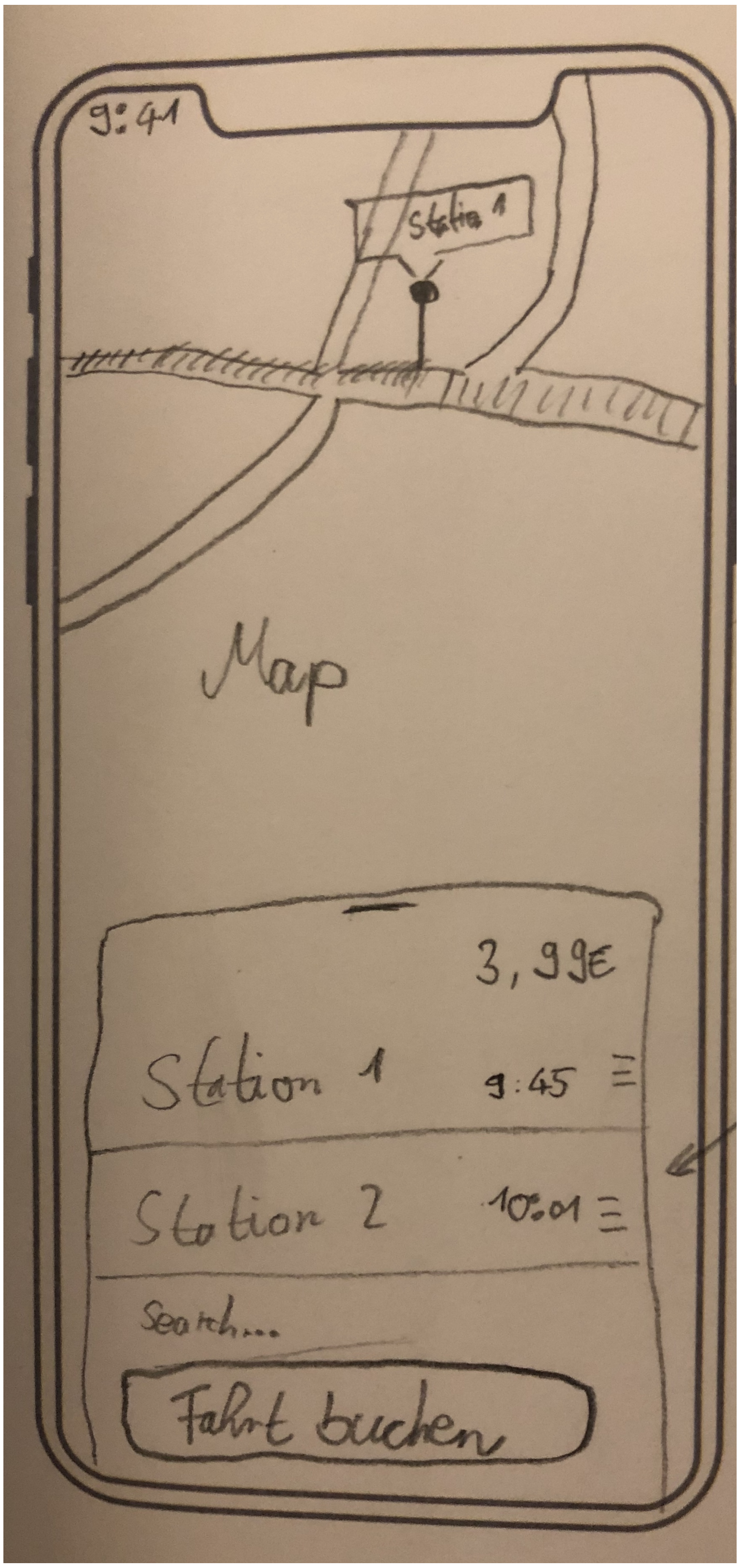


Figure 4.1: The first paper prototype featured the basic elements of a map and a planning tool.

tween stations and arrival time per station. We want to find a way to provide all this information to them without putting too much cognitive load onto the user, especially when they might not be interested in planning accurate times at that time. A prototype showing all the information can be seen in Figure 4.2.

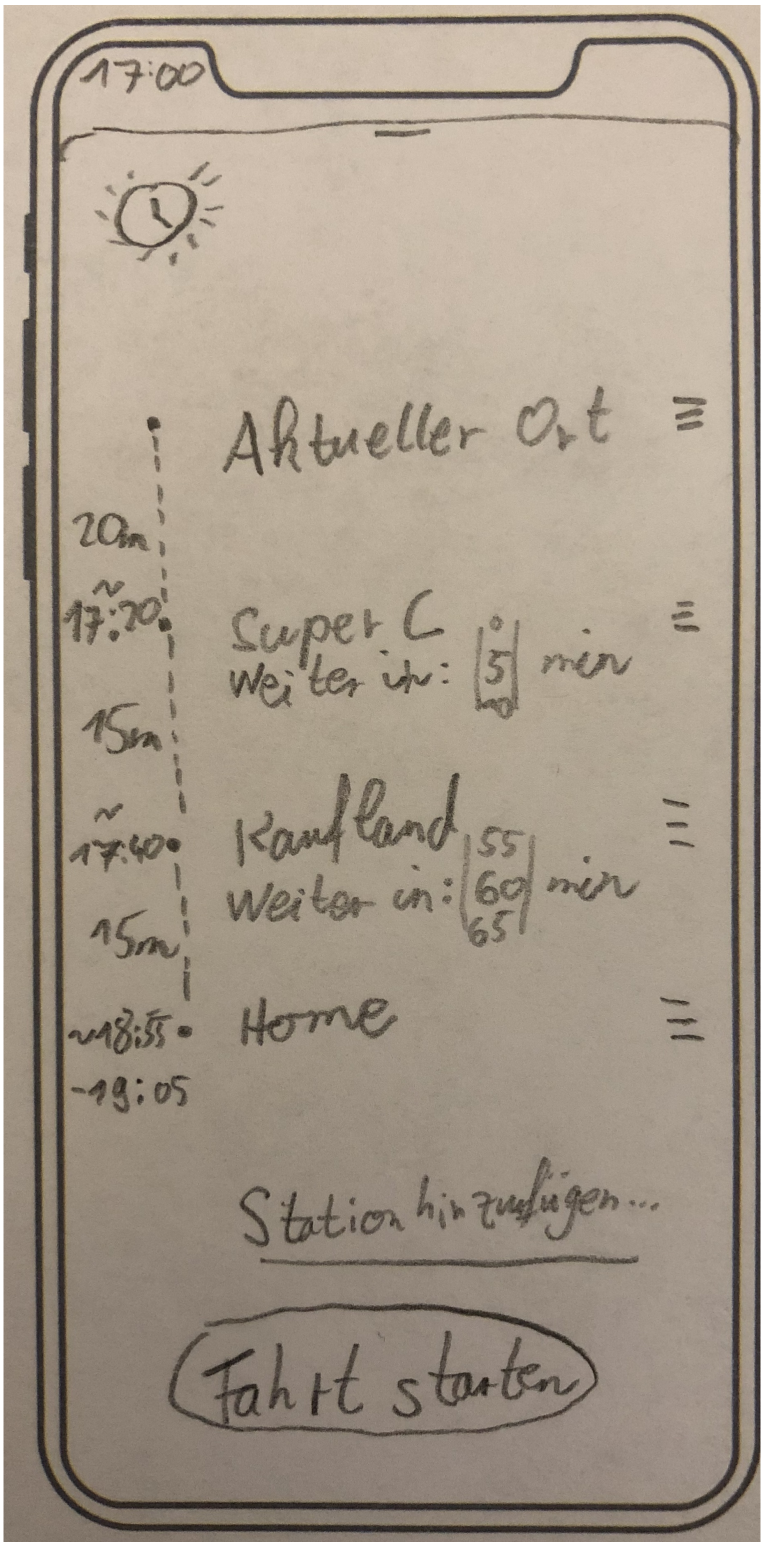


Figure 4.2: This prototypes shows all information required to accurately plan the duration of a trip.

4.3.1 Testing the need for a complex time planning.

Users questioned if time planning is needed. We performed a small user test.

We received some Feedback from users questioning if all the information for planning a trip is really required. Some users say they would prefer to only see the travel time from station to station and estimate the total duration themselves.

Two prototypes, one with and one without time planning.

We build two paper protoypes, one showing the complex planning feature (including arrival times and stay time) and one only showing the duration between stations. We used prottapp.com to make our prototypes interactive. Using Prottapp, we were able to scan our paper prototypes and define tappable areas that transition to a different scan (see Figure 4.3). This way users understand very well that this is not a fully-functional app (Because they see paper drawings) but can still easily navigate the prototype.

On a side note, in our very first interactive paper prototype we had a feature switched on which highlighted the tappable areas when the user tapped outside a tappable area. We had to disable that feature, because users quickly learned to tap anywhere on the screen and then one of the highlighted areas, completely bypassing the goals of the test.

We constructed a scenario which was impossible to fit into the time constraints.

We performed our test between-subjects using the following scenario:
It's 5pm. You need to go to the Super C to throw an attest into the mailbox (5min) and you need to do your groceries at Kaufland (30min). You need to be at home for dinner at 6pm.

There are now two possible routes the user can choose: Current Location → Super C → Kaufland → Home (which takes 50min) and Current Location → Kaufland → Super C → Home (which takes 30min). We already selected Super C at the beginning of the test to "trick" users into taking the longer route initially. It's intentionally not possible to perform the trip and be back home at 6pm.

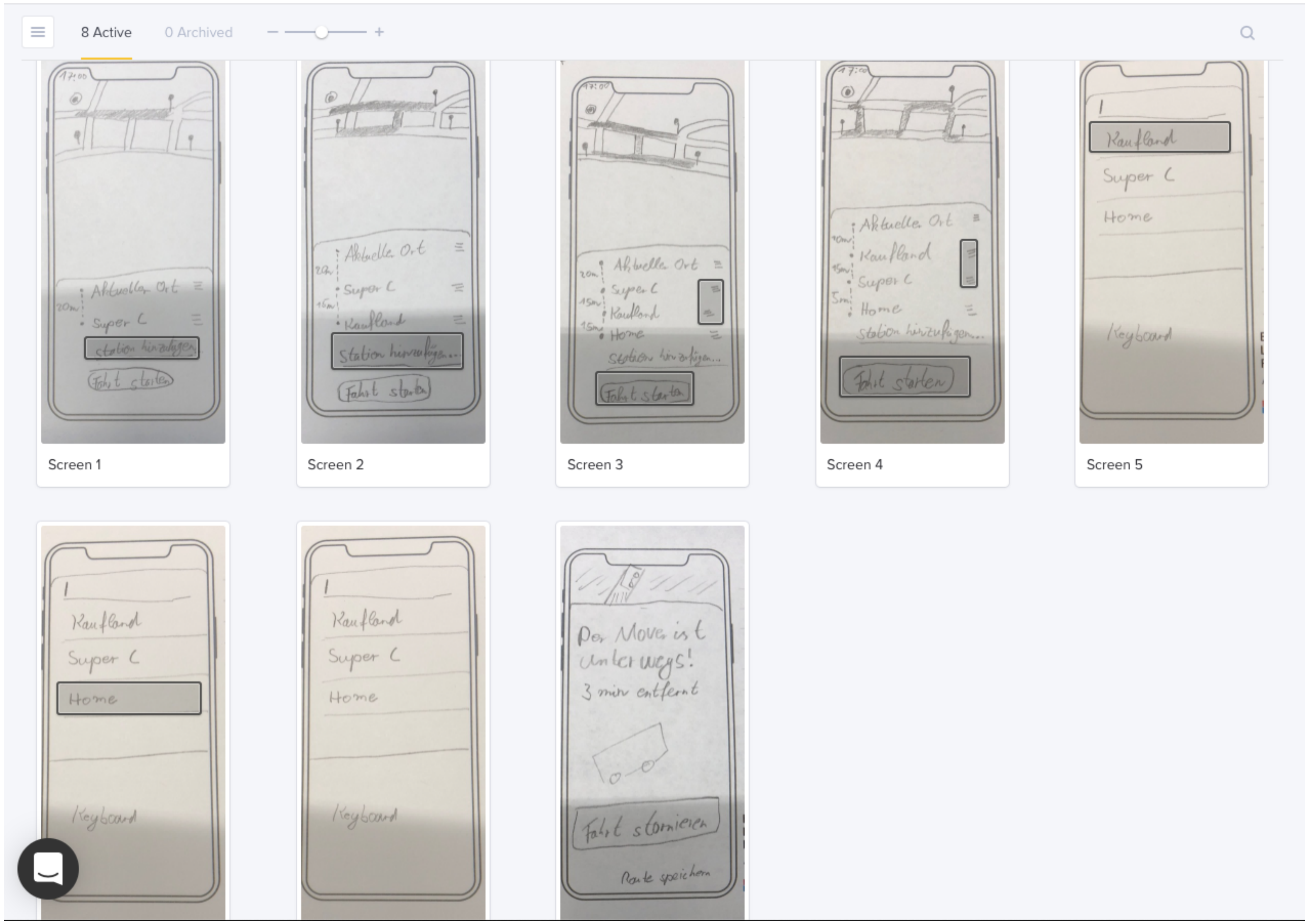


Figure 4.3: Prottapp allows us to create interactive paper prototypes. The marked areas can be tapped and transitions to other screens can be defined. The prototype shown here does only show the time between stations. The other prototype was similar, but showed the additional info as shown in Figure 4.2.

The users with time planning detected that the trip doesn't fit their schedule.

We observed, that users add the stations correctly and detect that the longer route does not fit their schedule. After reordering to get the shorter route the users without the complex planning feature felt confident that they can now perform the trip within the time frame. Only after asking if they are really sure they took some time to calculate all times and came to the conclusion that it doesn't fit their schedule. All users using the complex planning feature immediately detected that their route still doesn't fit into the schedule.

We concluded, that there is actually a need for a better route planning interface. Nevertheless, we shouldn't discard the users concerns that for some use cases the information might be irrelevant or to prominent. Because of this we decided in our software prototypes that we want to test hiding and showing the time planning features depending on the context.

4.4 Software Prototype

Because of the growing complexity in our prototype we decided to move on to a software prototype. Only in software we are able to dynamically add, remove and reorder stations. We want the user to be able to add arbitrary stations to their route and interact with the map or planning as they which.

We build an iOS app to get more dynamic behavior into our prototype.

We decided to start with a native iOS prototype (as opposed to Sketch or different tools) to get the level of dynamic behavior we need. Furthermore, this way we can iterate on our prototype until we get a finished project without a need to recreate elements.

4.4.1 Bottom Sheet

We build a bottom sheet as open source component.

We started by implementing the bottom sheet. We developed the bottom sheet as a separate framework, which can be found at `https://github.com/iCarambaa/STIBottomSheet`.

The bottom sheet features a center view controller, which will later hold the map in the background and an arbitrary number of bottom sheets can be added on top of it. We developed the bottom sheet as a custom container view controller. When a view controller is added it gets wrapped into a parent that adds a grab indicator at the top and optionally a close icon. The parent insets the view controllers safe area to make room for those elements. This way, it is possible to add any `UIViewController` as a bottom sheet without any further customization needed.

To allow easy transitioning from the minimized to maximized bottom sheet we add a pan gesture recognizer to the sheet and search for a `UIScrollView` instance in the bottom sheet. In minimized mode panning the bottom sheet always transitions to maximized. If the sheet is maximized, we decide dynamically if the `UIScrollView` should be scrolled or if the bottom sheet should move. We try to keep the behavior of the bottom sheet similar to the behavior in the Maps app to use the law of experience.

4.4.2 Map

We use MapKit for the map.

We use `MapKit` for showing the map. `MapKit` is built into iOS and easy to use, we can use the Map, the Annotations and `MKDirections` to calculate a route between two stations. The route can easily converted into an overlay to be shown on the map.

4.4.3 Planning

We built a custom collection view layout for the planning UI.

The planning UI is the most complex in our prototype. We need to show a list of stations, add arrival and duration times, we want to show a stay time for the stations and allow reordering. We decided to build this as a custom ‘UICollectionViewLayout‘.

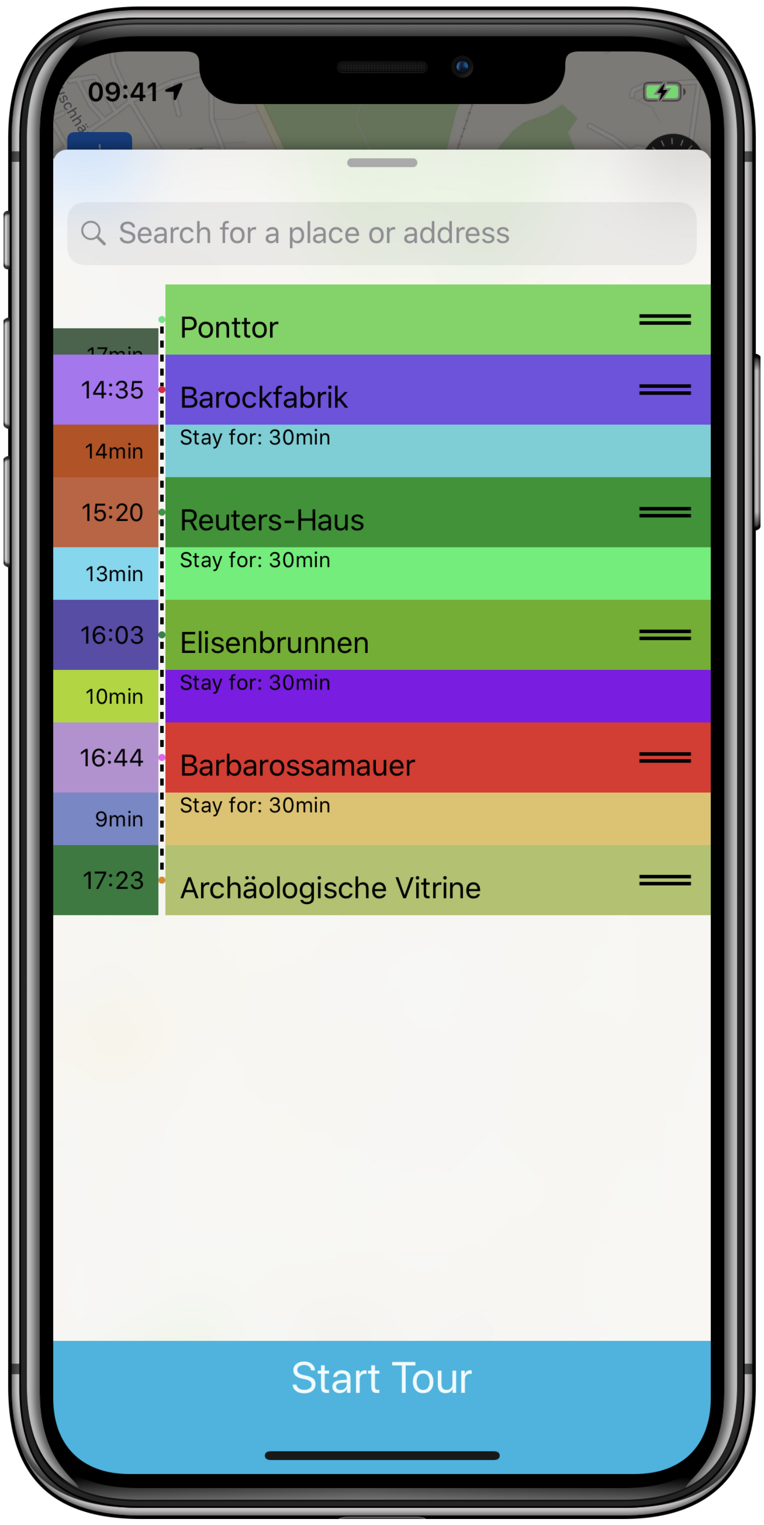


Figure 4.4: Here you can see the individual parts of the layout in different colors. We make extensive use of supplementary and decoration views in order to keep the individual parts of the layout small.

The layout can be seen in Figure 4.4. We make extensive use of `UICollectionViews` supplementary and decoration views to split the layout into small reusable parts. By having separate views for the content of the cells and e.g. the arrival time we can change our layout without needing to change our views or our datasource. Also notice, that the travel times and line decorations are not directly related to a cell, but are rather related to a pair of cells (As they describe the time between station A and B). With supplementary and decoration views we are able to interpret the index as the nth pair of stations which fits our data nicely.

Furthermore, our layout is notified when bounds change, that makes it easy to invalidate parts of it when transitioning from a minimized to a maximized sheet. We can use this to show a more compact view when minimized.

After some iterations, we concluded that the stay time is best set in the info sheet.

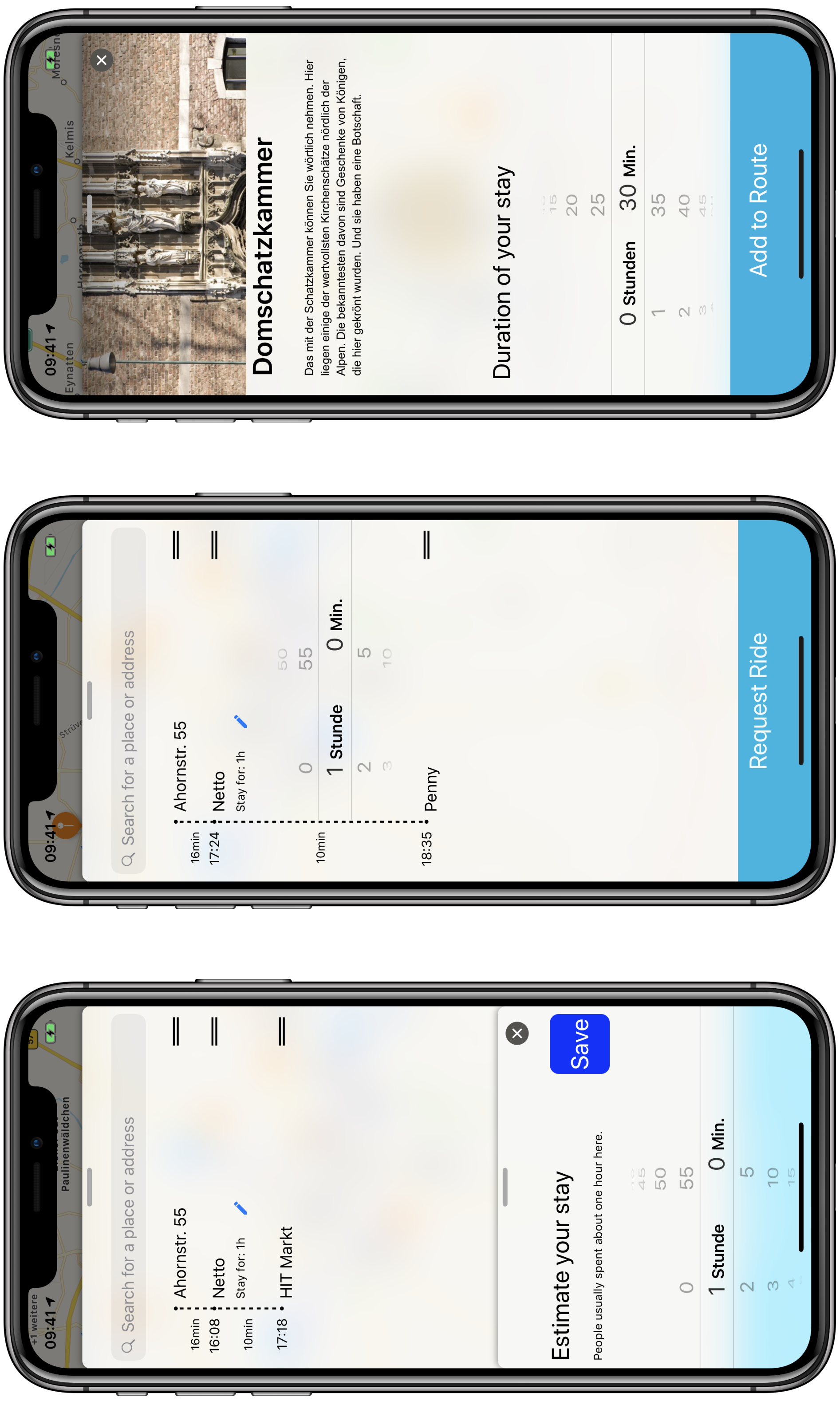
We performed many iterations on how to set the stay duration (see Figure 4.5). At first, we thought about adding an additional sheet to edit stay times. This also had the benefit that we could easily place information on how long people usually spend at those stations to the sheet.

We realized that users tend to get confused when we use to many sheets. Therefore, we tried to set stay durations directly inline (second screenshot).

4.4.4 Points of Interest

We built a flexible framework to add new POIs.

We want to show Points-of-Interest (POI) on the map. We have a subclass of `POIProvider` for every type of POI we want to show. They are combined into one `POIProvider` and have a method `poisInRegion(_:)` that returns all POIs in a given `MKCoordinateRegion`. That method works (API-wise) similar to `-[UICollectionViewLayout layoutAttributesForElementsInRect:]`. You have to return all POIs that are in the given region but can also return additional POIs. For the prototype, we simply return all POIs we have stored.



Das mit der Schatzkammer können Sie wörtlich nehmen. Hier liegen einige der wertvollsten Kirchenschätze nördlich der Alpen. Die bekanntesten davon sind Geschenke von Königen, die hier gekrönt wurden. Und sie haben eine Botschaft.

Figure 4.5: Iterations from left to right. We ended up combining the stay duration and info sheet.

We have some simple POIs like Supermarkets or ATMs, but also extracted the POIs used in the Aixplorer developed at the chair[Aix]. We decide what annotation view to use on a per POI base. All POIs except the Aixplorer POI use the default Marker, the Aixplorer POIs use the custom image markers used in the Aixplorer.

Each POI can be associated with a view controller that provides additional information. For the Aixplorer POI, we have taken the view controllers from the Aixplorer App and integrated them here.

We reused the content from the Aixplorer.

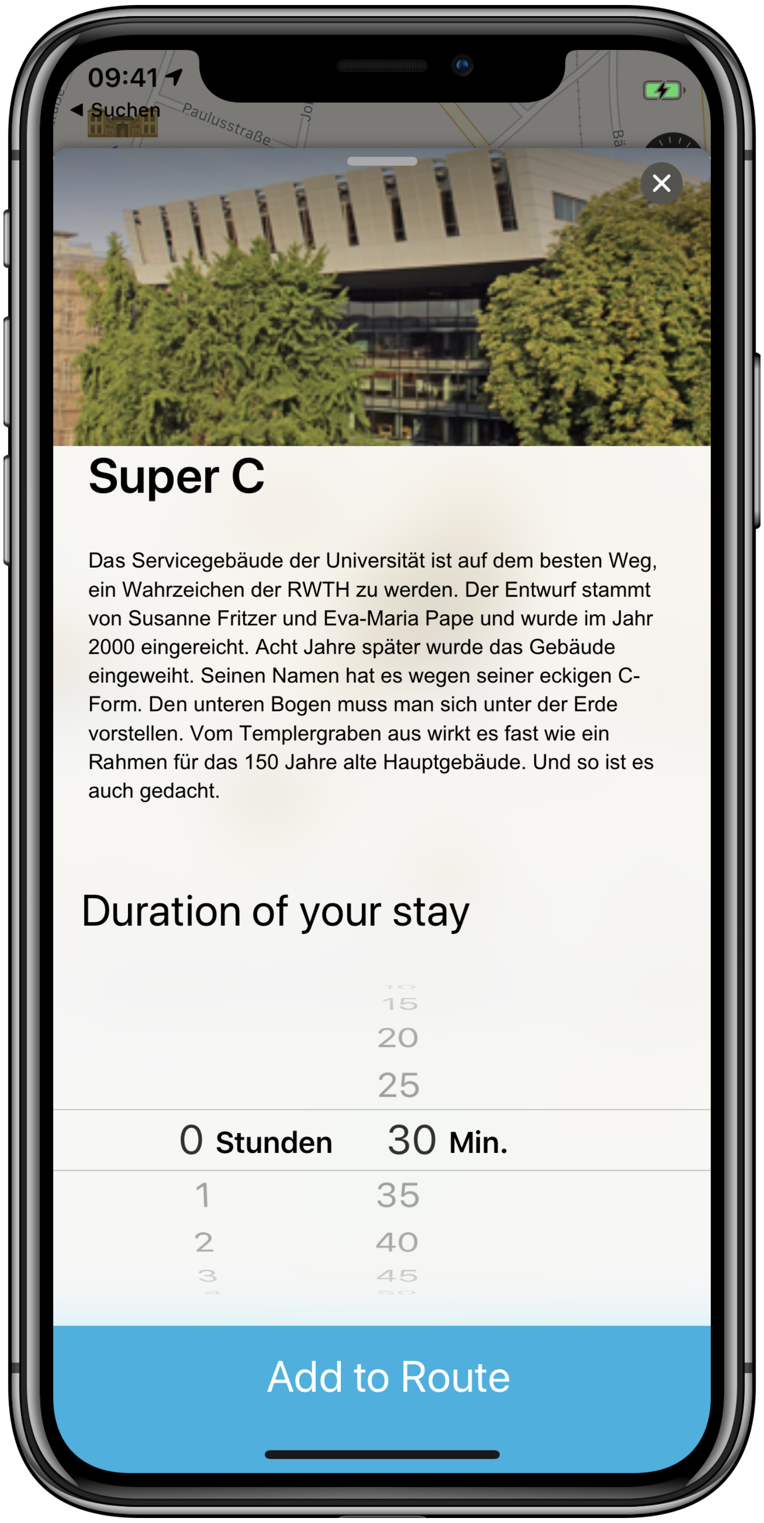
4.4.5 Architecture

In our app we have quite a few view controllers on-screen at the same time. There is the bottom sheet controller, the map, the planning, and info sheets for POI stations. One of the key requirements of our architecture is that we are able to make big changes without having to many dependencies in our code. We decided to build the app using a MVC-C (Model View Controller - Coordinator) architecture [Zabłocki][Khanlou]. Every view controller is designed to be independent of it position in the app flow. View controllers are created by a Coordinator and get their data and/or dependencies from the Coordinator. View controllers do not perform any navigation logic on their own but talk to a delegate, which is usually the coordinator.

We built the app using the MVC-C pattern.

We have a class `AppCoordinator` which mediates between the map, planning and info sheets. Take the following example: A user taps on a station on the map. The `MKMapView` notifies the `MapViewController` that the station has been selected. The `MapViewController` notifies its delegate, the `AppCoordinator`, that a station has been selected. Now, the coordinator is responsible for deciding what happens in that case. Depending on the feature flags (our final prototype had about 40 feature flags), the coordinator might decide to show an info sheet for that station or to tell the `PlanningViewController` to highlight the selected station. Assume the coordinator decided to show an info sheet. Now when the user

The AppCoordinator mediates between the different view controllers.



Super C

Das Servicegebäude der Universität ist auf dem besten Weg, ein Wahrzeichen der RWTH zu werden. Der Entwurf stammt von Susanne Fritzer und Eva-Maria Pape und wurde im Jahr 2000 eingereicht. Acht Jahre später wurde das Gebäude eingeweiht. Seinen Namen hat es wegen seiner eckigen C-Form. Den unteren Bogen muss man sich unter der Erde vorstellen. Vom Templergraben aus wirkt es fast wie ein Rahmen für das 150 Jahre alte Hauptgebäude. Und so ist es auch gedacht.

Duration of your stay

Figure 4.6: When a POI is selected we open an info view controller in a separate sheet. The view controller is wrapped into parent view controllers that provide the ability to set a stay duration and to add the POI to the route.

closes that sheet, the coordinator is notified and tells the `MapViewController` to deselect the selected station. By encapsulating the navigation logic (especially with all those feature flags) inside of coordinators we were able to keep our view controllers clean and simple.

Chapter 5

User Study

One of the key questions in the app is how users can associate stations on the map and in the planning. Once there are about five stations added to the route it is no longer easily perceivable, what stations you have to reorder in order to get a shorter route. The first thing we did to tackle this problem was to add a numbering to the marker annotations on the map. Another thing we tried (in addition) is to highlight stations in the planning (using color) when the stations is selected on the map and to select the stations on the map when the station is tapped in planning. We found that this behavior makes it easy to find a station, but it conflicts with the info sheet, which was usually shown when a station is selected.

How do users associate stations on the map and in the planning?

Due to the size of mobile screens, we can not show the complete route in a bottom sheet. Every cell needs enough height to show the name, the stay duration and labels for arrival and travel time. We looked into possibilities to reduce the amount of information depending on the context.

Can we hide information depending on the context?

5.1 Test Conditions

We put four different conditions under test. First, we want to analyze our idea of highlighting stations when the user

Condition C vs D compare what happens when the user selects a station.

selects them. In condition C, we show the info sheet whenever a station is selected, either on the map or in planning. In condition D, we highlight the station in blue when selected and don't show the info sheet. Only when the user taps an already highlighted station (either on map or in planning), we show an info sheet. We call this a double select.

Condition A vs B compare showing planning information only when maximized or always.

The other two conditions are related to showing and hiding information based on context. Our assumption is, that users will probably relate stations in planning and annotations while the sheet is minimized. We assume they will maximize the sheet in order to perform actions like reordering or planning on it. Therefore, in condition A the planning features are only shown if the bottom sheet is maximized. That means, if it is minimized, stay duration and arrival times are not shown (But travel time is always shown). This way, more stations are visible and we assume that this makes optimizing a route easier. In condition B all info is always shown. There are less cells visible when minimized, but the discoverability of the planning features is better.

5.2 Setup

We had four test groups. The study was performed between-subjects.

We performed two scenario tests with each tester. The first one aimed at planning a new trip and the second one aimed at modifying an existing route. Both scenarios were performed between-subjects. We had two times two conditions, therefore we formed four test groups. During the test we did a screen-recording, external video recording and audio recording. We used the Aixplorer POIs for both tests.

Testers were recruited from different projects at the e.Go office. We had 8 testers in total (5 male, 3 female) between 19 and 44 years (n=8, Mean=25.625, Median=22.5, SD=8.176). We had two testers per feature combination.

5.2.1 Scenario 1

For the first scenario, users were provided with the app after a fresh start. They were given the following scenario:

The first scenario was about planning a route.

"You want to plan a trip along the sights of Aachen which takes three hours in total. You want to start at Marienkirche and end at Annakirche."

Testers were expected to add a number of stations to the route and to use the search for the specific stations Marienkirche and Annakirche. They need to check the time planning info to make sure the route takes three hours. For this task its optional to adjust the stay duration because we wanted to check if users start setting durations on their own. We made this explicit in the second scenario.

5.2.2 Scenario 2

For the second scenario, we loaded a prepared route into the app (see Figure 5.1). The stations are added in a zig zag order which makes the travel times very long. Testers were given the following task:

The second scenario was about modifying an existing route.

"Change the route in a way that makes sense to you. Plan two hours for lunch at Reuters-Haus."

We setup the task vague on purpose to observe what users will do. The most obvious thing is to reorder the stations in a way that minimizes the travel times, but it's also possible to group them by topics or in a way that makes your route pass streets with other interesting stations. We asked them to plan two hours for lunch to make sure they need to change duration at least once.

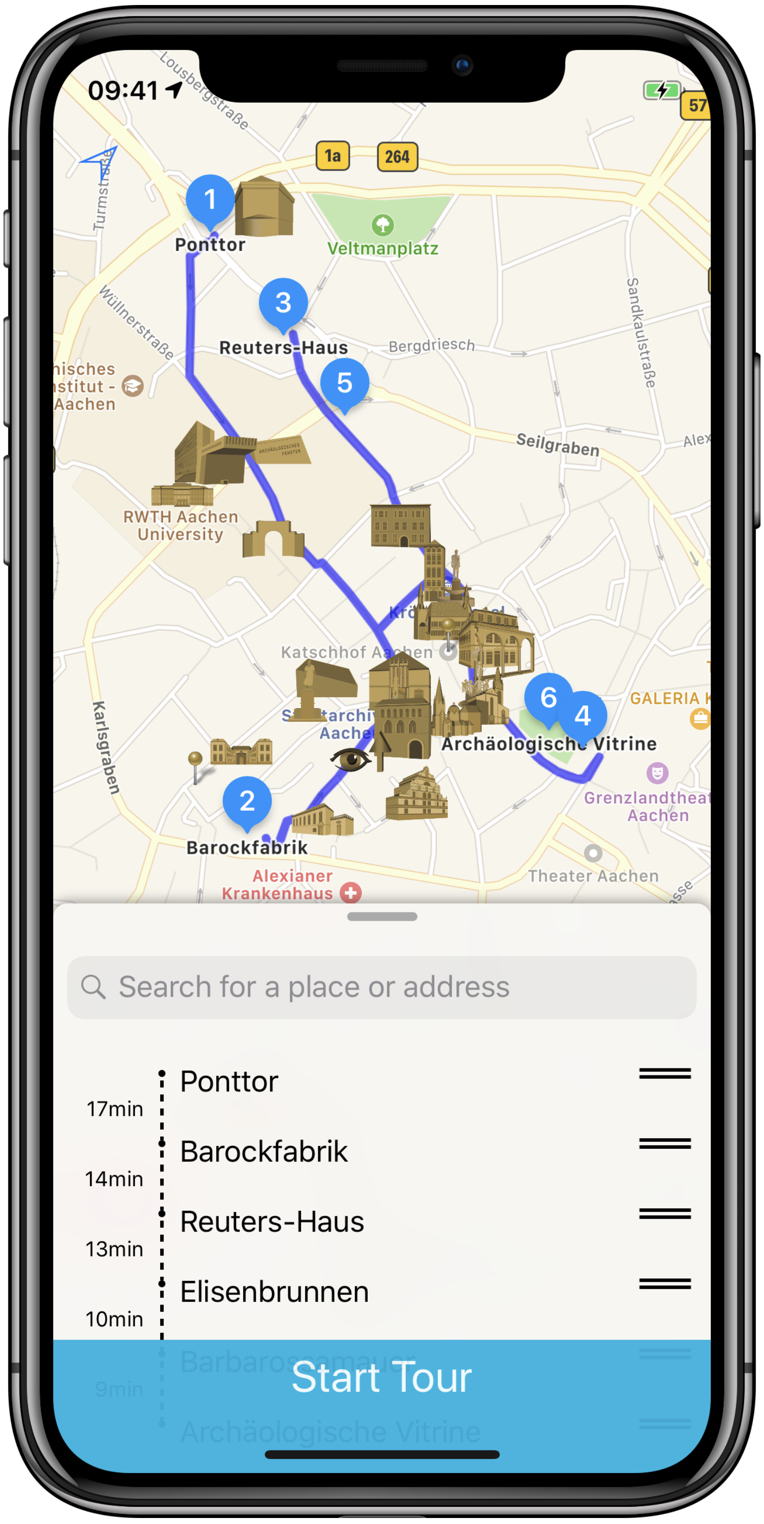


Figure 5.1: This route as prepared for the second scenario. It's a long zigzag route. The testers were asked to bring the stations in a more useful order.

5.3 Categorization

Categorization was done in ChronoViz. We used automatic annotations and manual observations.

We categorized the results by annotating them in ChronoViz [Fouse]. ChronoViz is an app that lets you combine multiple videos and add annotations on a timeline (see Figure 5.2) While the user used the app we recorded some events which we exported as annotations for ChronoViz automatically. The main reason for the automated annotations is record events that make orientation in the recording easier. For example we annotated maximizing/minimizing the bottom sheet, reordering stations or using the search. We also annotated the selection of the predefined route for the second scenario which made it easy to find the start of the second scenario in the video.

We added a second timeline for the manually added annotations. We added annotations in five categories: Success, Error, Gulf, Software Bug and other (or default).

While annotating we took additional notes for every feedback and observation we got.

In the success category, we annotated when the user first used a feature. Separately, we noted if there was any trouble detecting or using the feature. For example, when a user started the first scenario and immediately opened the search we annotated ″Success Search″ and wrote down ″Used search confidently.″.

In the Error category we tracked errors like ″Swipe to delete″ or users tapping the grab handle which is intented for reordering.

The Gulf [Norman, 2002] category is for situations, in which the mental model of the user fundamentally collides with the system model.

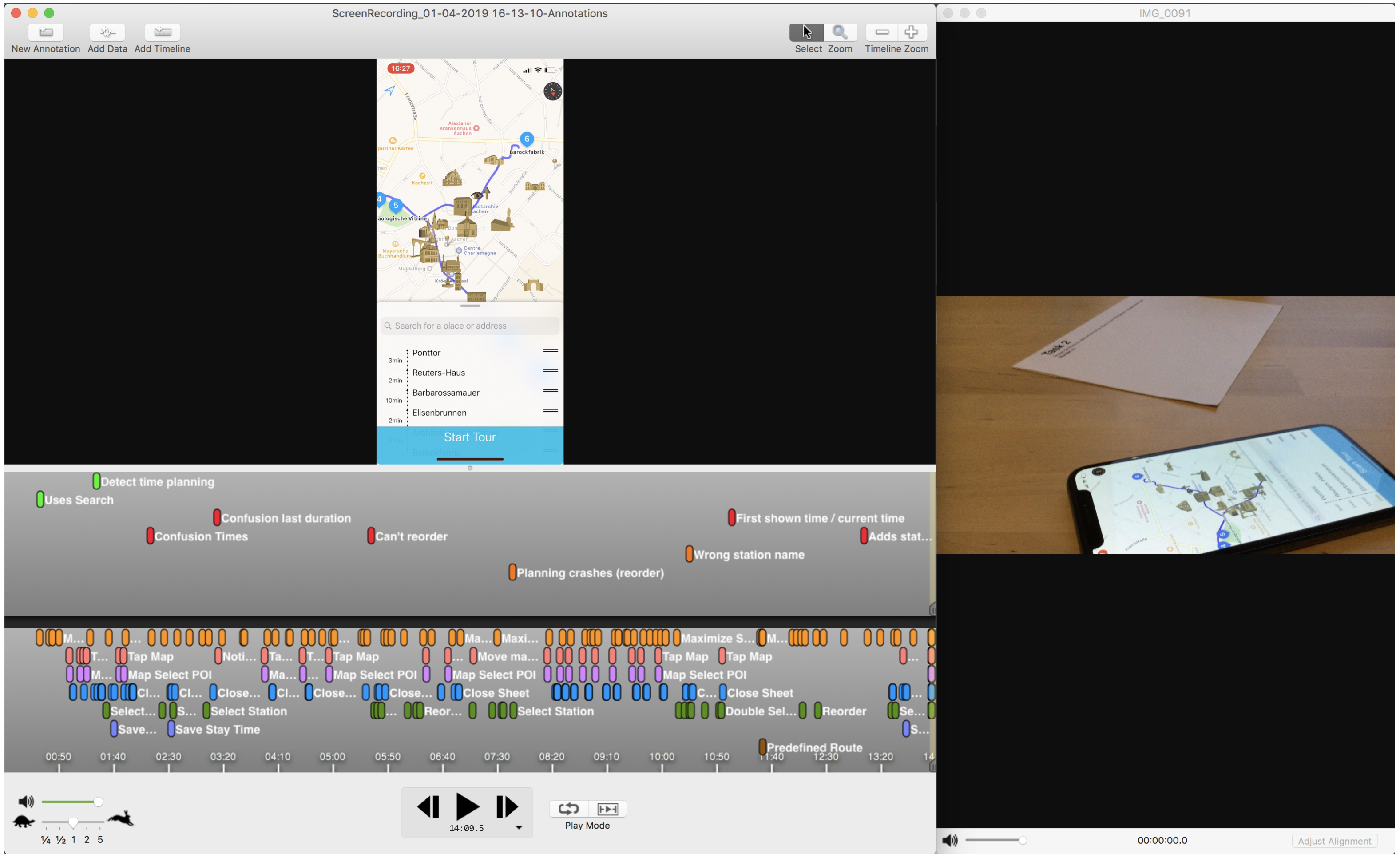


Figure 5.2: Videos are annotated in ChronoViz [Fouse]. We have a number of annotations that are automatically tracked in the bottom timeline and manually entered annotations on top.

5.4 Observations

5.4.1 All Groups

We received much feedback which was not related to the test conditions.

Sometimes, users were confused by the "Start Tour" button. Some wondered, if they need to tap that button in order to start planning a tour.

Most users use the search bar to find the Marienkirche. Most users started by exploring the stations on the map. Few of them didn't notice the search bar but kept searching for the Marienkirche on the map.

Three users asked if they could search for a station without adding it to the route or got confused, that the station was added to the route after selecting in search.

Only one user selected a predefined route. One user mentioned, that the predefined routes look like additional stations.

Many users tap the reorder icon to open the info sheet. Many try to reorder stations by dragging from the center of the cell. Some also try to remove stations using a swipe gesture on the cell.

Users are likely to minimize the planning by panning on the indicator, but minimize and maximize the info sheets panning anywhere on the sheet. Sometimes users invoke the notification center when trying to minimize the planning.

Often, users were unable to evaluate their total trip duration, often confusing start time and arrival time at second station. Users frequently said that seeing their total trip duration would be nice.

5.4.2 Condition A

The planning features were hard to find for users.

Users have difficulties finding the planning features. They move the bottom sheet often just enough to see all stations (but don't maximize completely). Some users said that they actively disliked the hiding of the planning features. 3 out of 4 testers confused the start time of their trip with the arrival time at the second station.

5.4.3 Condition B

Testers with condition B use the planning features.

Every user detected and used the planning features. They minimize/maximize the sheet more often in order to reorder stations, but didn't complain about it. 3 out of 4 testers understood the planning correctly, one confused start time and arrival time at second station.

5.4.4 Condition C

Users are able to associate stations on the map with stations in the planning.

Users associate stations by counting manually or by memorizing names of stations. No user complained that the info sheet is in their way. No user showed issues relating stations on the map to stations in planning.

5.4.5 Condition D

It's surpising for users when the info sheet doesn't appear after selecting a station.

Users were frequently surprised that the info sheet didn't open, especially when they wanted to set durations. Usually, they reacted as if their touch "hadn't worked" and tried again with some delay, also multiple times. We asked some users after the test what they think happens when they tap a station and they were not sure and "guessed" that the info sheet opens.

Only one user understood the behavior immediately and double tapped the station on the second try. Unfortunately, that didn't work in our prototype, because it requires two separate taps (not one double tap) but was nevertheless an interesting observation.

5.5 Discussion

In our user study, we learned a lot about the features we explicitly tested, but also about the other features, that were the same for all test groups.

Some feedback we received in every test group was differently strong for different test groups. Therefore, we will discuss the general feedback first and highlight differences when discussing the individual features.

5.5.1 Discussion of General Feedback

Starting a Tour

We should hide the "Start Tour" button until the user created a route.

The "Start Tour" Button is confusing, especially when no stations are entered yet. It's not really clear what the button does and if users have to tap in order to plan a route. The button could be hidden if less than two stations are added. In our test, the user had no context what is intended to happen when one "starts a tour". We assume that users who downloaded a "tour guide" would have a mental model which closer matches our expectations.

Search

The two users who didn't find the search bar seem to have gotten lost exploring stations. Because they had info sheets open all the time, the search bar was covered and not detectable. Both users came to a point, where they noticed the search bar, one stated that it was "very obvious" and "he has been stupid (sic!)". We believe that this is only a minor problem on the first use and users will very quickly learn how to use the search.

Stations shouldn't be automatically added to the route.

It was common that users wanted to search for a station without actually adding it to their tour. A station should only be selected, but not added in the next prototype. This behavior would also be similar to Apple Maps, which also shows a station, but doesn't start a navigation right away.

Our predefined routes aren't very discoverable right now and look like regular stations. We should probably add some section headers to indicate that there is a section for routes and one for stations.

Reordering

Reordering using a reorder control isn't a general learned concept.

Surprisingly, the reorder icon isn't widely recognized. Most users assumed they could reorder stations by panning on any point of the map. Furthermore, they tapped the reorder icon in order to open the info sheet.

Small details can make reordering better.

We looked again into how `UITableView` handles reordering and noticed that there is a small delay between `touch down` and being able to reorder the cells. We could do the same here on our cells: Wait for a longer touch down, then start reordering the cell for reordering. That way a quick swipe up would still result in moving the bottom sheet. Looking at the drag and drop introduced in iOS 11 it seems like a good platform-fit to allow drag and drop on the complete cell (nevertheless this is still just reordering and still different from the iOS 11 drag and drop which is app or systemwide).

We would need another test if users still find the reordering feature if the icon is gone.

Removal of Stations

Removing stations worked well. Swipe-to-delete could be added.

Generally, users found the option to remove a station very quickly. One user mentioned they were confused that the button to remove a station looked the same as the button to start the tour. Maybe a different color could help here. Some users tried to remove the station using a swipe-to-delete gesture, which could be implemented in the planning.

Total Trip Duration

The most common issue evaluating the total trip duration was that the start time wasn't explicitly shown, but just assumed to be the current time. Therefore, the arrival time was also not displayed on the first station. Users often falsely assumed, that the first time they saw, the arrival time at the second station, would be their start time.

We had a Gulf evaluating the total trip duration.

It would be better to show the arrival time at every station, including the first one (and same for the duration).

Showing the arrival time at every station might help.

5.5.2 Condition A vs B

In Condition A and B we compared always showing the planning features (Arrival time, Stay duration) in planning (B) vs. hiding them in the minimized view and only showing them when the sheet is maximized (A).

We found, that users are likely to miss the planning features when they are hidden. We couldn't see any better performance reordering the stations or relating annotations to planning stations. Users seem to get frustrated having to maximize and minimize the sheet.

Users with condition A were more likely to confuse different times shown when calculating their total trip time (3 out of 4 compared to 1 out of 4).

Users commonly opened the sheet just enough to see the information they need, which enabled very quick interactions, but by using the app in this way they rarely maximize the sheet completely.

As a result, the planning infos should always be visible.

Planning features should always be visible.

5.5.3 Condition C vs D

In Condition C and D we analyzed what should happen, when a user selects a station, either on the map or in the planning.

We didn't see any of the conditions perform significantly better than the other. Highlighting the station instead of showing the info sheet was intended to make the association between map and planning easier, but we didn't see any improvements due to this change.

The info sheet should always appear.

Because no user complained that the info sheet is disrupting him and few had trouble finding the right station we concluded, that showing the info sheet whenever the user selects a station is the right thing to do.

Adding numbers to the stations would help users.

Quite a lot of users (in both groups) said that a numbering of the stations would be helpful, so this is something we can do instead.

Chapter 6

Location-Based Content Push

We've already seen in the related work that dynamic, context-aware content is an important aspect for a tourist app. If we can monetize an app with content the user actively wants it's a win for both the user and the app publisher.

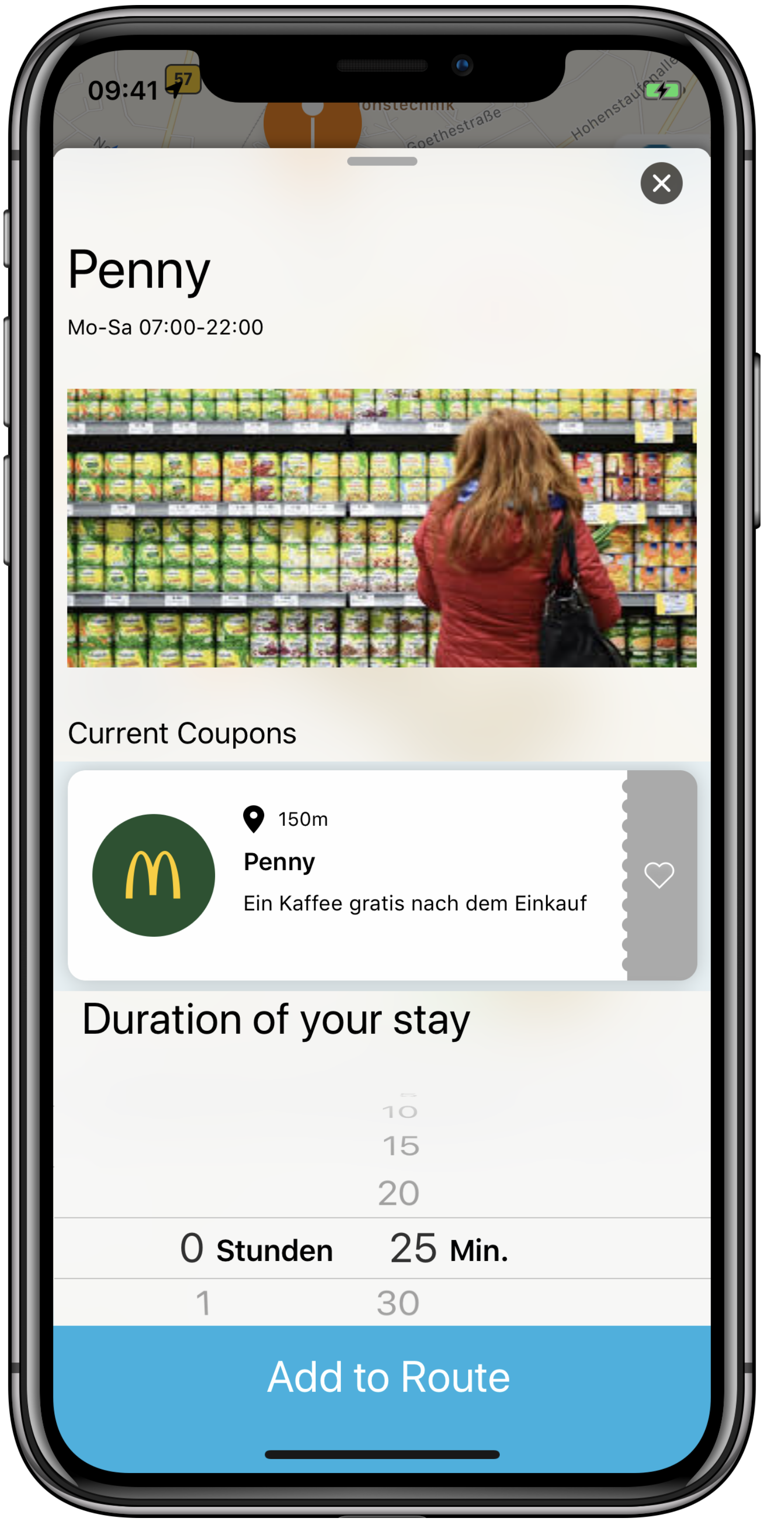
There are many sorts of offers that might be interesting for a user. Imagine a user wants to visit a museum, and he finds two museums that he finds interesting. If one of the museum has a special offer for the user available, the user is likely to chose that museum. The user and the museum would benefit. When the user actively searches for something, we can even show him offers we wouldn't relate to that user otherwise, see Figure 6.1 for an example.

Different categories of content could be relevant to a user.

6.1 Diamond Offers

If the user is interested in a specific POI, looks into it and finds an offer that's great, but there might also be offers the user wants, but doesn't search for them or even know they exist. A theater for example might add an offer where they heavily discount the last tickets for the evening.

Special Offers could be added to the map.



Penny

Mo-Sa 07:00-22:00

Current Coupons

150m

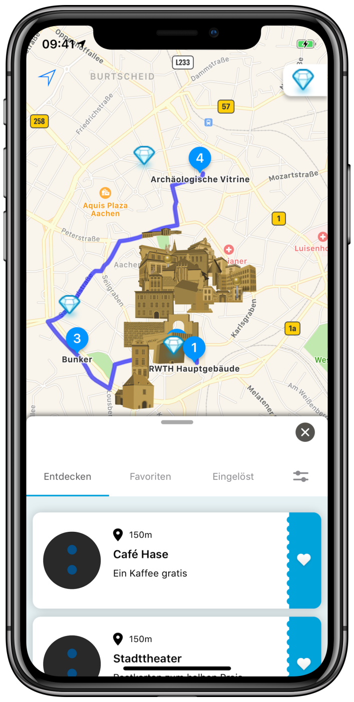
Penny

Ein Kaffee gratis nach dem Einkauf

Duration of your stay

	15
	20
0 Stunden	25 Min.
1	30

Figure 6.1: The info sheet providing useful information about a station could also provide the user with useful offers. When the user actively searched for a supermarket, he is probably interested in offers at that supermarket. Image is a mockup.



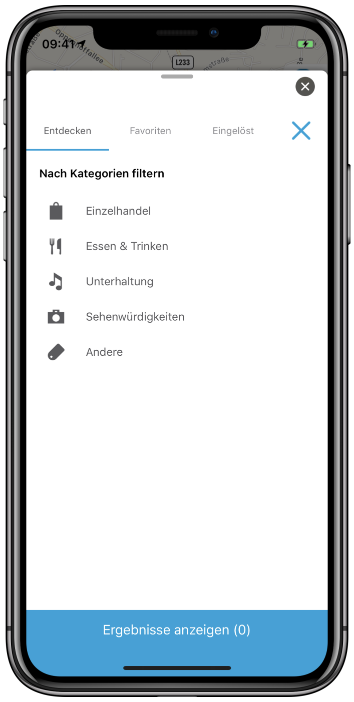


Figure 6.2: On the left are the diamond offers shown. The offers are related to the users route and shown on the map as well. Users can save offers to their favorites to redeem them later. Optionally, users can also filter for categories (right).

A tourist who is currently doing a sight seeing trip in the area probably wouldn't want to miss that.

A button opens offers in a bottom sheet.

We added a small button on the top right which shows a diamond icon and can optionally be badged when new offers are available (see Figure 6.2). When the user taps that button, we open a bottom sheet showing all the current offers, related to the users route. Only when the user explicitly opens the diamond offers diamond annotations are added to the map to show the locations. Users can save offers they want to use to their favorites in order to redeem them later. If they select an offer in the list or on the map they can get additional information.

6.2 Implementation

We implemented the Location-Based Content Push as a separate library because it might be useful in many different applications. We called the SDK XRoads and split it into three Modules: XRoadsAPI, XRoadsUI and XRoads.

6.2.1 XRoadsAPI

XRoadsAPI is responsible for fetching content from the server and used to provide the server with the necessary information to perform an educated guess on relevant content. In order to do so, it receives a list of the currently selected stations and sends it - in our prototype - to a Mock-server.

The content has a category, a description, a title, details, and images. We tried to build a quite general framework for creating new coupons which is still specific enough to be displayed uniformly.

6.2.2 XRoadsUI

XRoadsUI provides all UI Components (Views and View Controller) used. It imports XRoadsAPI so that it can use the data structures created there. Like in the route planning app, we made extensive use of custom container coordinators to achieve maximum reusability. The collection view showing a list of coupons for example is a separate view controller embedded into the view controller that manages the different tabs. Because of that, we are able to reuse that view controller when we want to show coupons inside of the POI info sheet.

We use the delegate protocol in our view controllers.

We use a similar data flow as in the route planner app. Every view controller receives input either in the constructor or in a separate property (if it can be updated) and talks back using the delegate protocol.

Our coupon collection view controller knows nothing about its content. If content changes, for example because the users adds a station of changes the category, it gets a new list of coupons. We use DifferenceKit [Aoyama] to calculate the difference between the old and new data and update our collection view accordingly (compare Heckel [1978]).

We use DifferenceKit to calculate updates to our data automatically.

Notice that we don't provide a way to present the view controller in the library. Users of the library are able to show the view controller as they wish. In our case, we use STIBottomSheet, the same framework we also use for all other sheets, and show the offers in a bottom sheet.

6.2.3 XRoads

The XRoads module glues everything together. It provides a coordinator that provides a simple facade. It uses Disk [Rizwan] to store the favorites in a json file.

A coordinator is provided that glues all parts of the SDK together.

In the route planner app our `AppCoordinator` creates an instance of `XRoadsCoordinator` and sets itself as the delegate. It updates the coordinator with location data and shows its view controller when necessary. The `XRoadsCoordinator` informs the `AppCoordinator` about the offers it currently presents so the `AppCoordinator` can make sure they are also shown on the map.

Chapter 7

Summary and Future Work

In this chapter we want to conclude the thesis. We first give a summary of our contributions and will then look into how the work can be extended in the future.

7.1 Summary and Contributions

We analyzed if and how route planning could be improved, especially with multiple stations. Although there already exist many route planning app, very few offer advanced features to plan an entire journey.

Our interface helps to find a route that fits a schedule.

We showed that being able to set the duration for a stations can help a user to decide wether a trip fits into his schedule. We noticed, that the interface can quickly grow complex which we couldn't solve by hiding information based on the context.

Numbering the map is important.

Users are better at associating the stations on the map with the stations in the list as we expected. It turned out, that adding a numbering of the stations on the map was very helpful and numbering in the planning UI would be convenient.

Location-Based content provides value to the user.

Location-Based content can enhance the experience in a routing app. Providing the user with relevant offers can be something the user wants, especially in a tourism context. Info sheets are a good location to place relevant content.

7.2 Future Work

Our route planning could be part of a complete tourist guide.

A future work could put the route planning into the context of a fully functional tourist guide. It could combine algorithmically created trips with the option to modify them.

Relevant content must be found.

In this thesis we assumed that there is a system available that provides as with relevant location-based content. There is additional work required to determine what content would be relevant to a user.

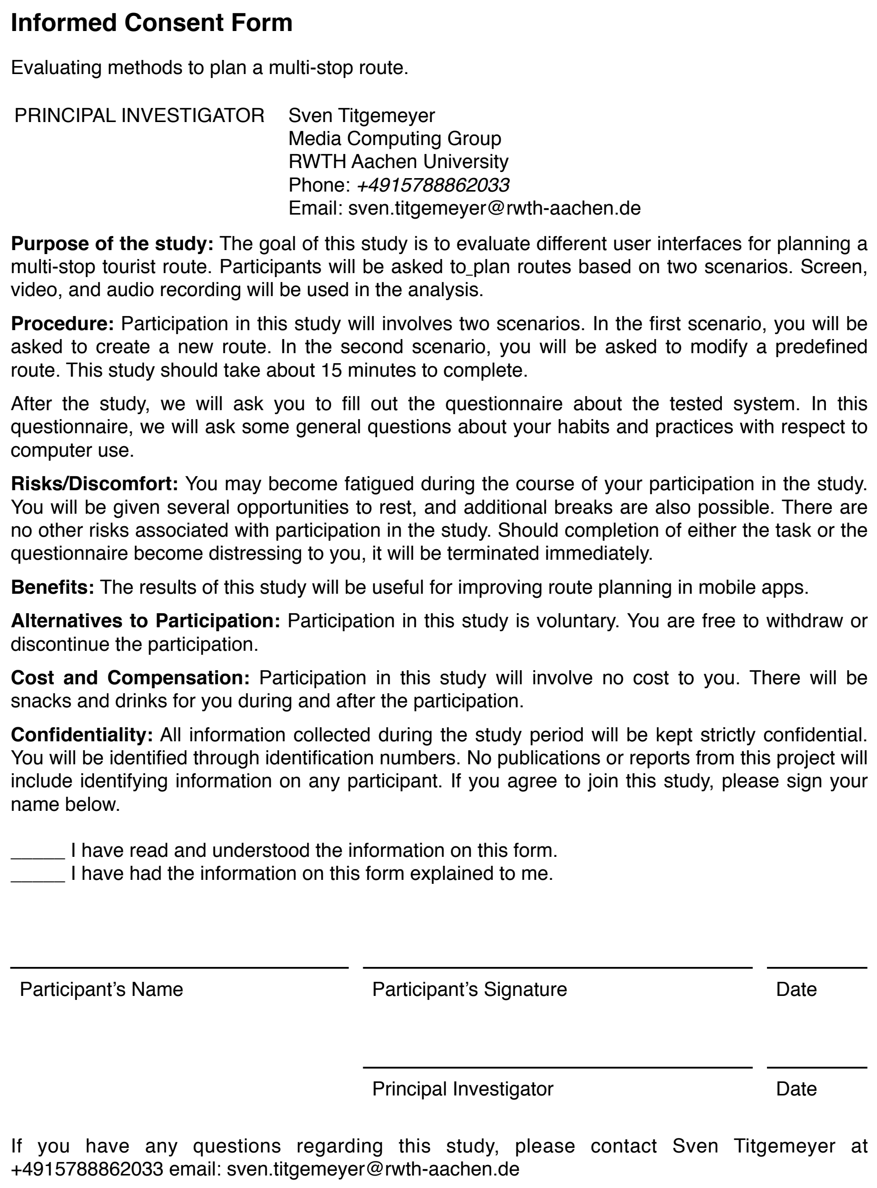
We plan to apply the research here to an On-Demand ride sharing app.

We plan to apply the work done here to an On-Demand ride sharing app in the near future. While we mostly looked into a tourist context here there is additional work required for day-to-day scenarios.

Appendix A

Informed Consent Dorm

The following informed consent form was used for the user study.



Informed Consent Form

Evaluating methods to plan a multi-stop route.

PRINCIPAL INVESTIGATOR	Sven Titgemeyer Media Computing Group RWTH Aachen University Phone: *+4915788862033* Email: sven.titgemeyer@rwth-aachen.de

Purpose of the study: The goal of this study is to evaluate different user interfaces for planning a multi-stop tourist route. Participants will be asked to_plan routes based on two scenarios. Screen, video, and audio recording will be used in the analysis.

Procedure: Participation in this study will involves two scenarios. In the first scenario, you will be asked to create a new route. In the second scenario, you will be asked to modify a predefined route. This study should take about 15 minutes to complete.

After the study, we will ask you to fill out the questionnaire about the tested system. In this questionnaire, we will ask some general questions about your habits and practices with respect to computer use.

Risks/Discomfort: You may become fatigued during the course of your participation in the study. You will be given several opportunities to rest, and additional breaks are also possible. There are no other risks associated with participation in the study. Should completion of either the task or the questionnaire become distressing to you, it will be terminated immediately.

Benefits: The results of this study will be useful for improving route planning in mobile apps.

Alternatives to Participation: Participation in this study is voluntary. You are free to withdraw or discontinue the participation.

Cost and Compensation: Participation in this study will involve no cost to you. There will be snacks and drinks for you during and after the participation.

Confidentiality: All information collected during the study period will be kept strictly confidential. You will be identified through identification numbers. No publications or reports from this project will include identifying information on any participant. If you agree to join this study, please sign your name below.

_____ I have read and understood the information on this form.
_____ I have had the information on this form explained to me.

Participant's Name	Participant's Signature	Date
	Principal Investigator	Date

If you have any questions regarding this study, please contact Sven Titgemeyer at +4915788862033 email: sven.titgemeyer@rwth-aachen.de

Figure A.1: The consent form was handed and explained to all study participants.

Bibliography

Aixplorer. `http://www.aixplorer.de`. Accessed: 2019-01-07.

Lyft. `https://en.wikipedia.org/wiki/Lyft`. Accessed: 2018-12-27.

Deutsche bahn. `https://en.wikipedia.org/wiki/Deutsche_Bahn`. Accessed: 2019-01-10.

Ryo Aoyama. A fast and flexible o(n) difference algorithm framework for swift collection. `https://github.com/ra1028/DifferenceKit`. Accessed: 2019-01-10.

Keith Cheverst, Nigel Davies, Keith Mitchell, and Adrian Friday. Experiences of developing and deploying a context-aware tourist guide: The guide project. In *Proceedings of the 6th Annual International Conference on Mobile Computing and Networking*, MobiCom '00, pages 20–31, New York, NY, USA, 2000. ACM. ISBN 1-58113-197-6. doi: 10.1145/345910.345916. URL `http://doi.acm.org/10.1145/345910.345916`.

Keith Cheverst, Keith Mitchell, and Nigel Davies. The role of adaptive hypermedia in a context-aware tourist guide. *Commun. ACM*, 45(5):47–51, May 2002. ISSN 0001-0782. doi: 10.1145/506218.506244. URL `http://doi.acm.org/10.1145/506218.506244`.

Adam Fouse. Chronoviz. `http://chronoviz.com`. Accessed: 2019-01-06.

Damianos Gavalas, Charalampos Konstantopoulos, Konstantinos Mastakas, and Grammati Pantziou. A survey on algorithmic approaches for solving tourist trip design problems. *Journal of Heuristics*, 20(3):

291–328, Jun 2014. ISSN 1572-9397. doi: 10.1007/s10732-014-9242-5. URL `https://doi.org/10.1007/s10732-014-9242-5`.

Hamed Haddadi, Pan Hui, and Ian Brown. Mobiad: Private and scalable mobile advertising. In *Proceedings of the Fifth ACM International Workshop on Mobility in the Evolving Internet Architecture*, MobiArch '10, pages 33–38, New York, NY, USA, 2010. ACM. ISBN 978-1-4503-0143-5. doi: 10.1145/1859983.1859993. URL `http://doi.acm.org/10.1145/1859983.1859993`.

Paul Heckel. A technique for isolating differences between files. *Commun. ACM*, 21(4):264–268, April 1978. ISSN 0001-0782. doi: 10.1145/359460.359467. URL `http://doi.acm.org/10.1145/359460.359467`.

Michael Kenteris, Damianos Gavalas, and Daphne Economou. Electronic mobile guides: a survey. *Personal and Ubiquitous Computing*, 15 (1):97–111, Jan 2011. ISSN 1617-4917. doi: 10.1007/s00779-010-0295-7. URL `https://doi.org/10.1007/s00779-010-0295-7`.

Soroush Khanlou. Coordinators redux. `http://khanlou.com/2015/10/coordinators-redux/`. Accessed: 2019-01-06.

Donald A. Norman. *The Design of Everyday Things*. Basic Books, Inc., New York, NY, USA, 2002. ISBN 9780465067107.

Katie O'Donnell and Henriette Cramer. People's perceptions of personalized ads. In *Proceedings of the 24th International Conference on World Wide Web*, WWW '15 Companion, pages 1293–1298, New York, NY, USA, 2015. ACM. ISBN 978-1-4503-3473-0. doi: 10.1145/2740908.2742003. URL `http://doi.acm.org/10.1145/2740908.2742003`.

Saoud Rizwan. Delightful framework for ios to easily persist structs, images, and data. `https://github.com/saoudrizwan/Disk`. Accessed: 2019-01-10.

Richard Schaller. Mobile tourist guides: Bridging the gap between automation and users retaining control of their itineraries. In *Proceedings of the 5th Information Interaction in Context Symposium*, IIiX '14, pages 320–323, New York, NY, USA, 2014. ACM. ISBN 978-1-4503-2976-7. doi: 10.1145/2637002.2637052. URL `http://doi.acm.org/10.1145/2637002.2637052`.

Pauline Schnor. Komoot-gründer: "man soll bei uns keine neuen freunde kennenlernen". `https://ngin-mobility.com/artikel/komoot-markus-hallermann-outdoor-app-mobility/`. Accessed: 2019-01-10.

Krzysztof Zabłocki. Improve your ios architecture with flowcontrollers. `http://merowing.info/2016/01/improve-your-ios-architecture-with-flowcontrollers/`. Accessed: 2019-01-06.

Index

Typeset January 10, 2019