

Artem Golovin

Personalized Journey Planner

Helsinki Metropolia University of Applied Sciences
Degree

Information Technology
Bachelor of Engineering

Thesis

15 April 2016

Author(s) Title	Artem Golovin Personalized Journey Planner
Number of Pages Date	42 pages + 0 appendices 15 April 2016
Degree	Bachelor of Engineering
Degree Programme	Information Technology
Specialisation option	Web Programming
Instructor(s)	Olli Alm, Senior Lecturer
<p>The goal of the GoAbout project was to build a personalized journey planner based on OpenTripPlanner, OpenStreetMap and a custom back end. The project aimed to implement personalization options to provide the best possible experience.</p> <p>The planner is mainly intended to be used in the Netherlands, however, it is based on open source solutions such as: OpenTripPlanner as a planning and routing solution, OpenStreetMaps as a map provider and ns.nl as a transit feed provider. It utilizes all three sources into one piece and puts a layer of personalization on top. The project can be adjusted for nearly any country which uses well-defined standards such as General Transit Feed Specification.</p> <p>The planner support user profiles and records user activity which is later used to suggest more relevant results. Also the project allows users to set their own or preferred types of transport and to filter out irrelevant options. The planner remembers previously made choices and adjusts new routes based on these choices.</p> <p>The main focus was on the personalization options within web application. It also consists of back end and a few other different front ends such as mobile applications and publicly placed touchscreens.</p> <p>The result is a ready to use ecosystem of products which allows users to plan routes in advance, check previously planned routes at any time in the future and to base new routes on the already planned ones.</p>	
Keywords	journey planning, navigation, personalization, OpenTripPlanner

Contents

1	Introduction	1
2	History of Journey Planning	2
2.1	Paper-based journey planners	3
2.2	Analog Navigation Systems	3
2.3	First GPS-based Navigation Systems	4
2.4	First Journey Planners	5
2.5	Current Journey Planners	6
3	Sources of Planning	10
3.1	Static Timetables (GTFS)	10
3.2	Real-time GPS-data (GTFS-realtime)	12
3.3	Maps (Street Networks)	13
3.4	Statistical Analysis (Custom Layer)	14
4	Personalization and Prediction Options	15
4.1	User's History of Movements	16
4.2	User's Preferred Types of Transport	17
4.3	User's Email and Calendar	18
4.4	Weather and Time the day	19
4.5	Popular Destinations	20
5	Structure of Prototype	21
5.1	Open Trip Planner	22
5.2	Back end	22
5.2.1	Geocoder and Geocoder	24
5.2.2	Trip Planner	26
5.2.3	User-specific Data	27
5.2.4	Reservations	27
5.3	Front end	28
5.3.1	Structure Overview	29
5.3.2	User Interface Overview	30
5.4	Summary	39
6	Discussion and Conclusion	39
	References	42

Acronyms and abbreviations

API	Application Programming Interface
CRS	Computer Reservation Systems
CRT	Cathode Ray Tube
CSS	Cascading Style Sheets
CSV	Comma Separated Values, a format for storing tabular data in a text file
CURL	A command line tool https://curl.haxx.se/
GDS	Global Distribution Systems
GPS	Global Positioning System
GTFS	General Transit Feed Specification
GTFS-realtime	General Transit Feed Specification in realtime
HAL	Hypertext Application Language
HREF	Hyper Reference
HTML	Hyper Text Markup Language
HTTP	Hypertext Transfer Protocol
IBM	International Business Machines
IT	Information technology
JS	Javascript
JSON	JavaScript Object Notation
LIFT	Locate, Identify, Flatten, Try
OSM	Open Street Maps
OTP	Open Trip Planner

REST	Representational State Transfer
TRANSIT	NAVSAT, Navy Navigation Satellite System
URI	Uniform Resource Identifier
USSR	Union of Soviet Socialist Republics
WWW	World Wide Web
XML	Extensible Markup Language
ZIP	Compressed File

1 Introduction

Due to the invention of computers, world wide web, email, messengers and different customer relationship management tools (CRMs), half of our activities can be done remotely: from home or any tropical island, we can submit our work in the web, we can order groceries and clothes online and meet our friends over Skype. However, in half other cases we still need to travel somewhere.

Journeys within a city were not that complicated in the middle ages when a man was able to cross the whole city in a few hours by foot. Modern cities can be up to 100 kilometers in diameter and due to many transport options a 21st century man can have a dozen alternative ways to get from point A to point B: Walking, cycling, taking a car, train, tram, bus or a plane and all their possible combinations. Journeys can be short 10 minutes rides or fairly long trips with 2-3 changes.

At some points, movements got so complicated that IT companies started to make services based on maps with special algorithms to plan journeys — where the user can choose a start point and end point to get to the destination in the fastest possible way. The first local planners such as the initial version of Reittiopas.fi (made in 2001) [15] only used timetables for route making but modern planners could also use real-time GPS-data from the public transport, traffic jams information, statistical analysis and user personal preferences to show the handiest way.

However, this is not enough. The next step is to predict the user's movements based on his/her past journeys and on the conditions such as weather, calendar, day of the week and others. Because every user is different: some prefer public transport, others prefer bicycles. Some users prefer to pay more to get there faster while others would choose the cheapest possible solution.

The thesis is based on a real experience gained while developing GoAbout personal planner in the Netherlands built on the top of already existing services such as OpenTripPlanner. The planner is publicly available on <http://www.goabout.com> and uses some of the ideas described in the report.

The author works with the GoAbout company on a daily basis as a front-end engineer, doing programming for all the web-based front ends such as goabout.com application and also for mobile applications such as swift-based iOS Planner. All the ideas and the user interface described in the thesis are common work of the core GoAbout team which consists of five members. Most of the implementation of the web-based front end was done by the author.

The GoAbout project itself is interesting in terms that a major part of population uses journey planners or GPS-navigators on a daily basis. However, most planners still use no or very little predictions and many other usability solutions already used by search engines, major web shops and in many other technological fields. These enhancements might be based on the person and include, for example, personalized search outputs based on previous input or might be factor-based and change output based on current traffic situation, weather or any other external factors, for example, hiding walk options when there is an icy shower outside.

The topic of the thesis is to show possible approaches for personalization and prediction options in a journey and to build a route planner prototype which includes a few of these approaches. The prediction is important because it saves the user's time and sorts journey options based not with the "fastest first" approach but with the "most convenient first approach". However, the meaning of convenient is different for every user so here is why personalization is also important.

2 History of Journey Planning

This chapter introduces the paper-based journey planners, analog navigation systems, the first GPS-based navigation systems as well as the early journey planners. Finally, the current journey planners with their extensive set of features are introduced.

2.1 Paper-based Journey Planners

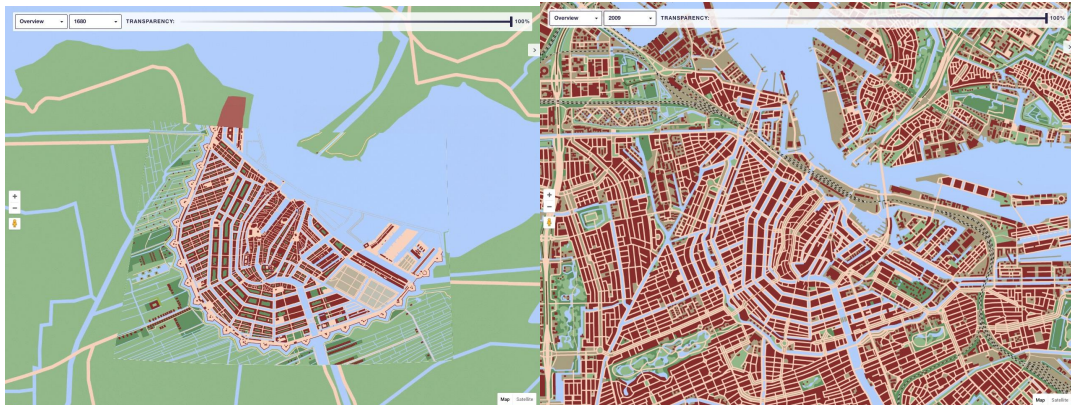


Figure 1: Maps of the City of Amsterdam in 1680 and in 2009. Both maps taken from <http://www.mappinghistory.nl/> [1]

Printed maps and timetables were the most popular journey planners in the world before the digital era. Cities were not that big a few centuries ago. For example, in 1680 Amsterdam was 5 kilometers long and 2 kilometers wide. It was easy to cross the whole city in two hours. So there was no need in Journey Planners, the whole city could be easily memorized in a few days. For example, Figure 1 shows two maps of Amsterdam in 1680 and in 2009. And as can be seen, the change has been drastic. [1]

Even bigger cities, such as London or Paris, started to have their own public transport systems only in the middle of 19th century. These public transport solutions were usually private and consisted of horse-operated omnibuses or small underground lines. The lines had no timetables and sometimes not even a strict route. [2]

2.2 Analog Navigation Systems

The first navigator in the modern form was introduced in 1920s. It did not use satellites or LCDs or anything else except little wooden rollers. Users had to turn them manually while driving along. The system was called Plus Fours Routefinder. It was supplied with about 20 different routes, including journeys inside London and even long trips like one from London to Edinburgh. The price of wristlet route navigator was 5 pounds (around 100 pounds compared to modern prices). Plus Fours Routefinder is shown in figures 2 and 3.



Figures 2, 3: A picture of the very first portable navigation system made in a form of wristwatches called Plus Fours Routefinder and it's in-car version Iter-Auto. Pictures taken from <https://habrahabr.ru> [13]

Later, the company introduced in-car navigation systems as shown in Figure 3. Those systems were bigger in size and some of them were synchronized with car's speed and were turning maps automatically. However, even a little extra turn would require users to set the map manually in the right position. [13]

These navigators never became popular because they only had a few selected routes and were complicated to use.

2.3 First GPS-based Navigation Systems

In the year of 1957 USSR launched the first satellite. American scientists watched the satellite over some time and realized that due to the Doppler effect, the frequency of the received signal was changing based on the satellite position in the sky. The closer he satellite was, the bigger was the frequency. It was possible to find the satellite position knowing the watcher's own coordinates and the other way around. If the position of the satellite was known, scientists were able to calculate their own position. [13]

And only three years after, the first satellite navigation system was launched by USA. It became operational on 13 April 1960. It was called “TRANSIT” and was mainly used to obtain accurate location information for ballistic missiles. It took almost 15 minutes each time to provide a single location reading. [3]



Picture 4: Etak Navigator. Pictures taken from <https://habrahabr.ru>[13]

The first public navigator became available in 1985. It was called The Etak Navigator, had a green CRT display and used magnetic tapes to read maps from. Etak Navigator is shown in Figure 4.

2.4 First Journey Planners

The first journey planners were made for airlines, to automatically suggest relevant flights and propose smooth transfers. They were initially called Computer Reservation Systems (CRS) and supported buying/reserving seats on the flight in the same time. Later, CRS were renamed to Global Distribution Systems (GDS) as soon as they started to support more than one airline. [14]

These systems were used to connect airlines and travel agencies in real time, so travel agents were able to find the best flight and instantly reserve seats, preventing overbooking or underbooking. The first system was invented in 1964 by America Airline and IBM and it was called Sabre. [14]

However, GDS were available only for travel companies and the systems were processing flights, not local in-city journeys. The first local trip planners were developed by the owners of travel data, in other words, public transport providers made first local trip planners.

Uusi haku - [Ohjeet](#) - [FAQ](#) - [Palaute](#) [På svenska](#)
[In English](#)
[Taskuversio](#)

Omat paikat ja reitit ovat väliaikaisesti poissa käytöstä.

Syötä/valitse lähtöpaikka ja määränpää

Mistä	<input type="text"/>	Kartalta	Hakemistosta	Esim. Porkkalankatu 18
Mihin	<input type="text"/>	Kartalta	Hakemistosta	Esim. Pasila
Kello	11 : 55	<input checked="" type="radio"/> Lähtöaika	<input type="radio"/> Perillä	
Pvm	27 . 10 . 2001			

[Tarkennettu haku](#)

Tervetuloa!

Reittiopas ehdottaa reittejä kahden valitsemasi paikan välillä käyttäen pääkaupunkiseudun joukkoliikennettä.

Syötä lähtöpaikka ja määränpää tekstikenttiin. Paikka voi olla katuosoite, pysäkki tai paikannimi. Voit myös valita paikat kartalta tai hakemistosta. [Ohjeet](#)

[Reittioppaan etusivulle](#)

Omat reitit Poista

Ei tallennettuja. Voit tallentaa omia reittejä myöhempää käyttöä varten. Katso [ohje](#).

Omat paikat Kartalla Poista

Ei tallennettuja. Voit tallentaa omia paikkoja myöhempää käyttöä varten. Katso [ohje](#).

[YTV Liikenne](#) - [Aikataulut](#) - [Ajankohtaista](#) Powered by Novo Meridian

Picture 5: A screenshot of Reittiopas website as of 27.10.2001. Picture taken from Archive.org website. <http://web.archive.org/web/20011027085527/http://pathfinder3.meridian.fi/ytv/fi/>

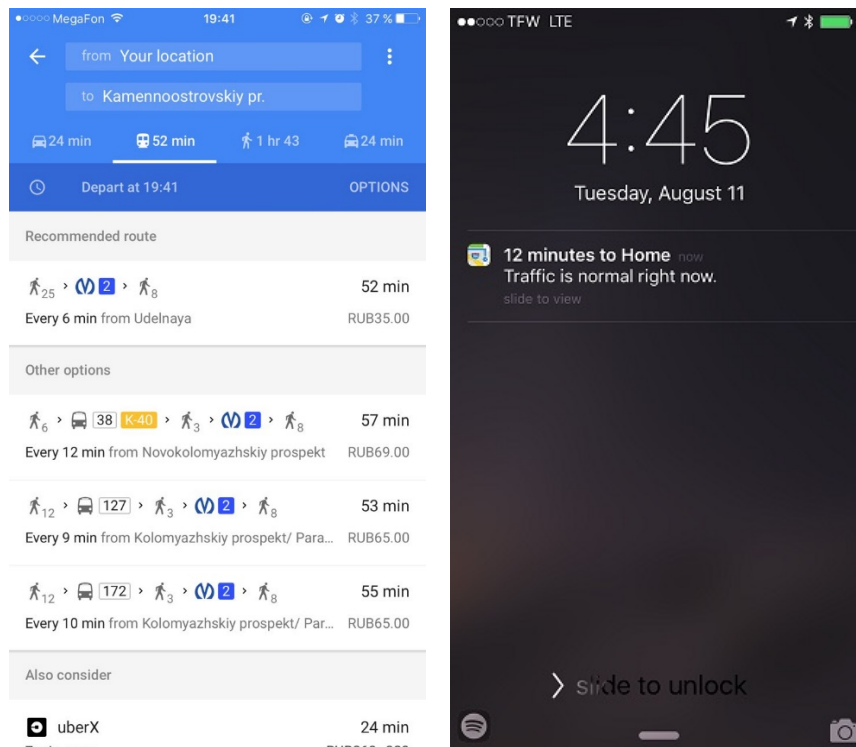
For example, one of the public first journey planners was developed in 2001 by the Finnish Pääkaupunkiseudun yhteistyövaltuuskunta (YTV, currently called Helsingin seudun liikenne, HSL). It is called Reittiopas and is still in use. Figure 5 shows the very first public version of Reittiopas. [15]

2.5 Current Journey Planners

Currently, there are many journey planners available around the world. Some of them are local (such as reittiopas.fi [15]), some are available almost worldwide (such as Google Maps). Some of the major players over Europe are Google Maps, CityMapper and Yandex.Maps.

Google maps works almost worldwide, it can plan car trips with voice guidance and turn-by-turn navigation everywhere, recognizes traffic jams and, most importantly, it has

public transport support almost in every city over the Europe and North America. However, Google Maps uses more or less the same approach worldwide and smaller local journey planners usually produce more accurate trip options since these planners are adjusted to local differences. Figure 6 illustrates planning functionality of the mobile Google Maps application.

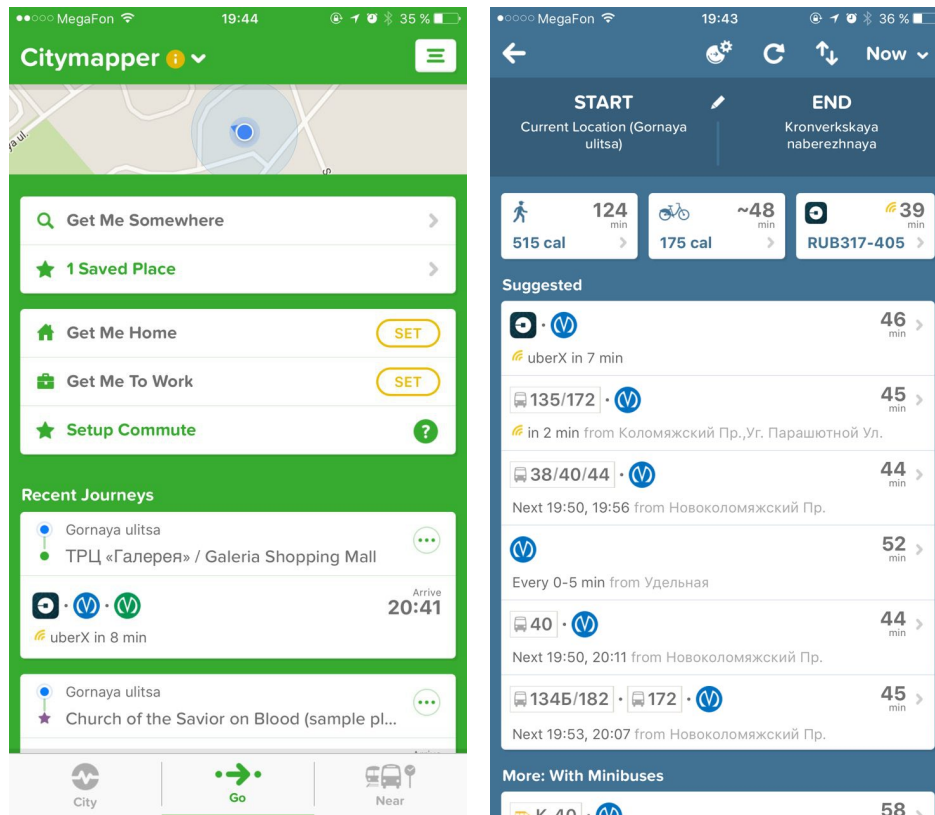


Figures 6, 7: Screenshots of Google Maps application and a push message sent by Google Maps application when the predicted trip starts

On the other hand, combined with “Google Now”, Google analyzes the user’s movements and tries to predict repeating trips. When the time comes, it automatically shows a push notification on the user’s phone with a destination, traffic info and approximate travel time. When swiped, Google maps are opened automatically with already planned journey. See Figure 7 for an example.

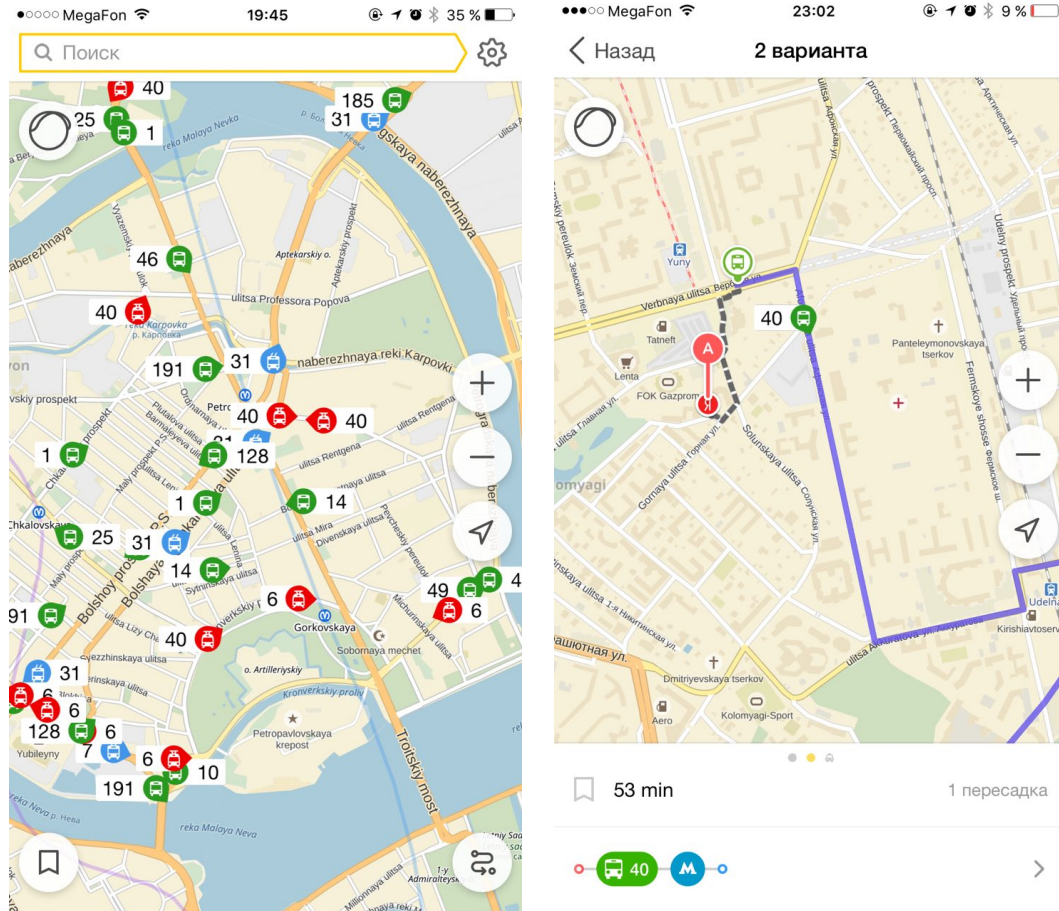
While Google Maps intends to cover as many cities as possible, CityMapper concentrates on capitals or megalopolises, about 25 cities in total, such as London, New York, Paris, Amsterdam or St. Petersburg. CityMapper gives a personal approach to each city, for example, Amsterdam and London are well known for its rainy days and these versions of CityMapper propose special “Rain safe” trip options. Also it has a number of useful hacks like showing which car of a train is the most convenient choice.

The mobile version of CityMapper automatically shows approximate travel time to the last used destinations as shown in figures 8 and 9.



Figures 8, 9: Screenshots of CityMapper application. Home screen and a journey screen

Yandex.Maps does not support the whole of Europe but it is a major player in Russia, ex-USSR countries and Turkey. The most notable thing about Yandex.Maps is that it is mostly realtime-based instead of timetable-based. Traffic jams are a serious issue in Russian cities, car accidents or other harmful events often take place there and produce unpredictable traffic jams around the cities. Yandex built a network of road sensors, webcams and, most importantly, users to quickly detect fresh congestions and to re-route drivers right away.



Figures 10, 11: Screenshots of Yandex.transport application. Home screen and a journey screen with the needed bus shown

As for public transport, timetables are not followed there so Yandex.Maps shows the actual position of the next bus instead of showing timetables as shown in Figure 11. In other words, Yandex takes local extravagance into account. However, it has zero personalization options and only recently started to keep a history of previously used destinations. [17]

3 Sources of Planning

This chapter introduces worldwide defined standards of defining timetables and realtime updates to them, the standards are called GTFS and GTFS-realtime, they make a solid base for every journey planner.

Maps might be in different formats, might be vector-based or raster-based, however, a map should have a layer with nodes (street objects) on top, otherwise it cannot be used with a journey planner since the planner uses nodes, not a picture, and the user sees a picture but the nodes are invisible for an end user.

3.1 Static Timetables (GTFS)

GTFS is a General Transit Feed Specification, it is a format for public transport schedules including geographic information. It is commonly used all over the world, including countries such as USA, UK, Australia, Finland or Russia. One common format allows trip planners use the same structure for different cities.

It is made of many text files collected in a single ZIP file. The text files usually include agency information, stops, routes, trips, stop times, calendar, fare attributes, fare rules and even route shapes. The standard specification requires text files called stops.txt, trips.txt, routes.txt, agency.txt, stop_times.txt, calendar.txt with (accordingly) stops, trips, routes, stoptimes, transport providers and a weekly schedule. Also ZIP file might include text files with fares, route shapes, transfers and other optional or region specific properties. Listings 1, 2 and 3 show examples of GTFS feeds. [7]

```

stop_id,stop_name,stop_desc,stop_lat,stop_lon,stop_url,location_type
S1,Mission St. & Silver Ave.,The stop is located at the southwest
corner of the intersection.,37.728631,-122.431282,,
S2,Mission St. & Cortland Ave.,The stop is located 20 feet south of
Mission St.,37.74103,-122.422482,,

```

Listing 1: Example of stops.txt in GTFS feed

```

route_id,service_id,trip_id,trip_headsign,block_id
A,WE,AWE1,Downtown,1
A,WE,AWE2,Downtown,2

```

Listing 2: Example of trips.txt in GTFS feed

```

trip_id,arrival_time,departure_time,stop_id,stop_sequence,pickup_type
AWE1,0:06:10,0:06:10,S1,1,0
AWE1,,,S2,2,1
AWE1,0:06:20,0:06:30,S3,3,0
AWE1,,,S5,4,0

```

Listing 3: Example of stop_times.txt in GTFS feed

Listings 1, 2 and 3 are typical examples of such text files inside a single ZIP file. The examples are similar to CSV tables and could be easily parsed.

Listing 1 is a list of stops, it consists of the required properties such as the stop id, stop name, stop latitude, stop longitude and a few optional properties such as the stop description or location type. [7]

Listing 2 is a list of trips. It includes route id, service provider id, trip id, trip headsign (direction) and might include optional properties such as block id (a block consists of two or more sequential trips made using the same vehicle, where a passenger can transfer from one trip to the next just by staying in the vehicle) or a shape id (shape defines how a line should be drawn on the map to represent a trip) [7]

Listing 3 is a list of stop times, it includes route id, arrival and departure times, stop id, stop sequence and might include optional parameters(?) such as pick up type (indicates

whether the stop will always happen or that passengers must notify the driver for a stop).
[7]

When stops, trips and stop times are combined in a trip planner, they provide enough data to make a route with. [7]

3.2 Real-time GPS-data (GTFS-realtime)

GTFS-realtime is an extension to GTFS that allows public transport providers send realtime updates about their fleet. It was made as a layer on top of GTFS to provide additional accuracy on transport movements.

Unlike regular GTFS, GTFS-realtime feeds are served via HTTP and updated frequently. In an ideal case, the feed is updated each time when a vehicle sends its new location or speed. The updates usually include trip information such as delays, cancellations and changed routes, vehicle positions and speed. Also it can include general service alerts such as moved stops, events affecting a station, route or the entire network. Listing 4 shows an example of GTDS-realtime feed.

```
entity {
  id: "3"
  trip_update {
    trip {
      trip_id: "frequency-expanded-trip"
      start_time: "11:15:35"
    }
    stop_time_update {
      stop_sequence: 2
      arrival {
        delay: 2
      }
    }
  }
}
```

Listing 4: Example of a GTFS-realtime feed in a JSON form to be human-readable, usually, that data is presented as a protocol buffer

It is worth noticing that GTFS-realtime feeds are provided in a completely different format. They are based on Protocol buffers which are language-neutral mechanisms of serializing structured data quite like XML but smaller and faster.

It is shown in a JSON form since encoded protocol buffer is not human-readable. The listing provides a realtime update that a trip with an ID of 3 on a stop sequence with an ID of 2 has a two minutes delay. [8]

The difference between GTFS and GTFS-realtime is that GTFS is provided once and only updated when some route/stop is permanently changed, it includes the timetables and all the related information and can weight a lot. While GTFS-realtime feeds are small and issued every few seconds, these feeds do not provide timetables, it provides a difference between real situation and a static represented in a regular GTFS feed.

3.3 Maps (Street Networks)

Street Network is a system of interconnecting lines and points that represent a system of streets or roads for a given area. A street network provides the foundation for network analysis; for example, finding the best route or creating service areas.

A base element of map structure is a node with geographic coordinates in it. The node can be anything: a house, a traffic light or a country, it can be a part of a line or of a relation. The node consists of a longitude, a latitude and a list key-value tags. Also maps consist of ways — sequences of nodes making representing, for example, roads, rivers, houses and relations — any type of relationship between nodes, ways or other relations. It is used to represent complex objects such as country borders or bus routes. Figure 12 shows how all three types of OSM objects are related. [9]

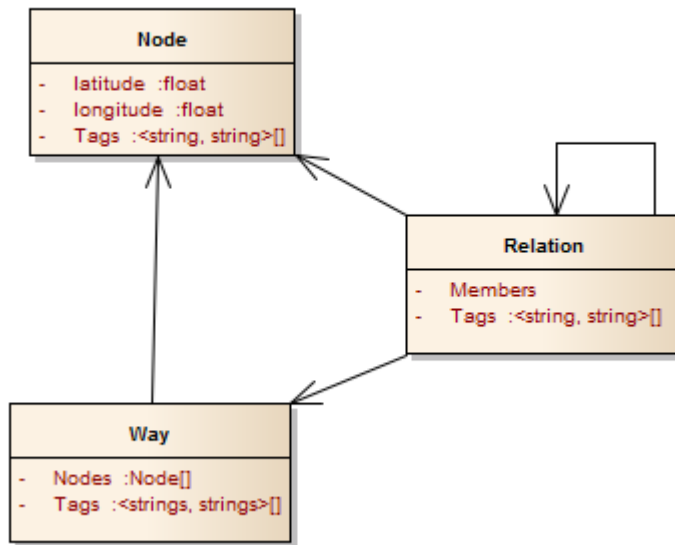


Figure 12: Basic objects in the OpenStreetMap data models

Altogether, nodes provide that additional layer of data which is parsed by trip planners and used for trip calculation. OpenStreetMap is an example of such a map and it is used in OpenTripPlanner. The map allows OpenTripPlanner to efficiently plan routes between nodes by analyzing the shortest Ways between them. [9]

3.4 Statistical Analysis (Custom Layer)

Statistical analysis is a custom layer on top of all the previous layers. There is no single industry-defined standard for analytics and moreover, that kind of data is usually a corporate secret of every trip planner.

It is used for improving planning accuracy, for example, during rush hours based on historically collected data. The data can be collected via GTFS-realtime feeds or by sending user's average speeds over specific routes. Figure 13 shows Yandex trip planner which has interface to look up possible future traffic jams based on already collected data for each road.

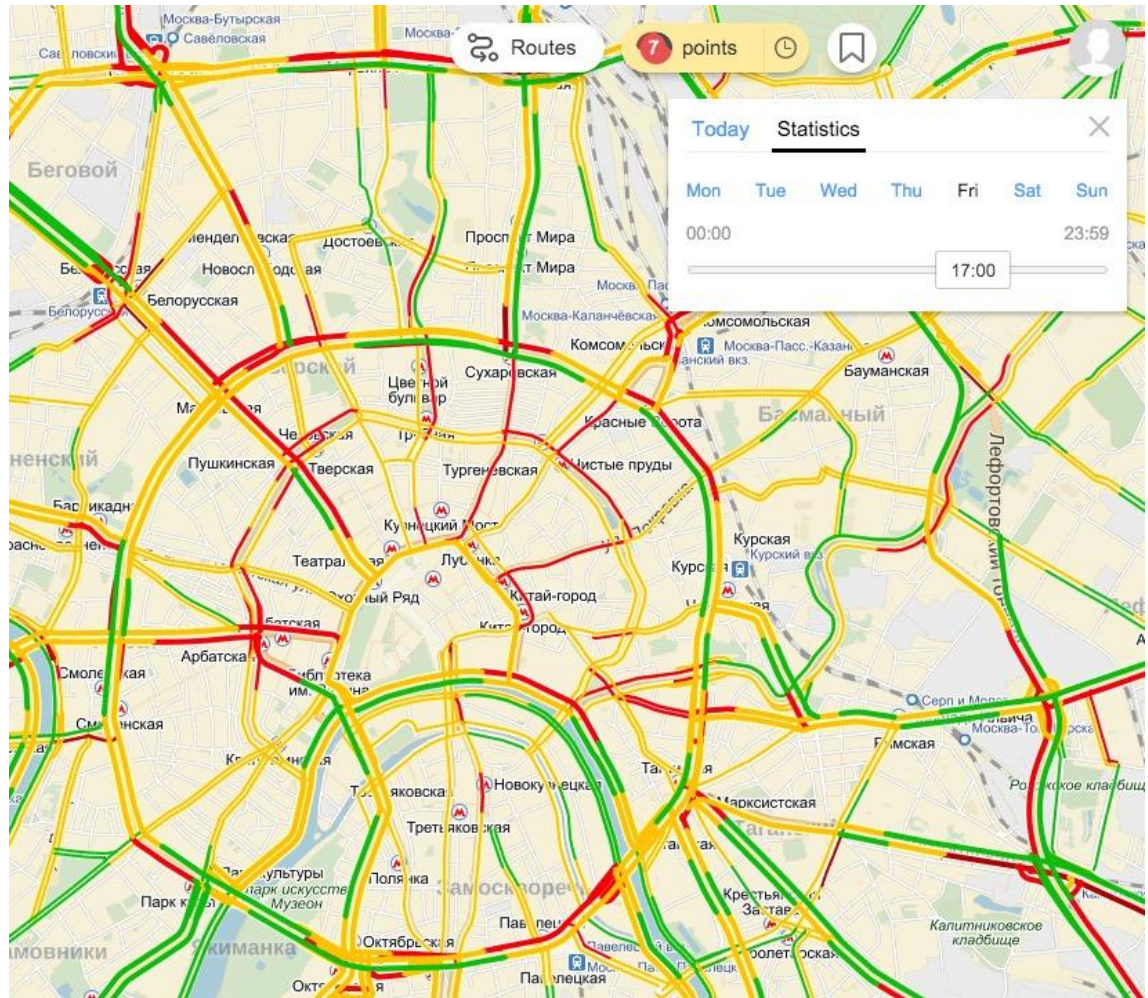


Figure 13: Yandex maps with statistics module. Predicts traffic jams using previously collected data.

That data is also used in trip plannings so it can affect the trip planner to change the route even if the selected road does not have traffic jams right now. With the given data, the trip planner can predict route delays or route changes and hence suggest longer travel times or completely different routes.

4 Personalization and Prediction Options

First of all, the main thing to mention here is that all the personalization and prediction options are put on top of an already advanced planner which solves the main problem of every planner — finding the quickest route. The planner already takes into account travel

time, amount of bus-changes, walking distance and probability of successful realization of the route. [10]

These personalization options are final steps to understanding user needs. It is not about analyzing the route and route algorithms, it is about finding what users want. In other words, one person likes long walks while another one would prefer to pay €10 for a cab. Some people prefer trains while others prefer buses. Some people do not care about weather while some others would prefer to leave their bicycle at home during rainy day.

4.1 User's History of Movements

One of the interesting personalization options is analyzing previous user trips and suggest new ones automatically based on the user's history.

Most people make the same trips at the same periods of time. For example, an average human being goes to work from Monday till Friday. Usually, the work starts at the same time and ends at the same time so it is quite easy to guess where he is going to travel on every weekday morning or evening. The same pattern can be predicted for many other movements: including getting groceries, lunches, picking up kids at kindergarten and so on. Even if the pattern is not so obvious, it can be established after some period of time.

The simplest way would be to use the same frequency as used in Mozilla Browser for calculating the most visited pages. "Frecency" is a combination of words "frequent" and "recent". The default Frecency value for all entries is 0, so first used values will always be displayed below visited places. Places visited in the last 4 days get a weight of 1 and will be on top, places visited in the last 5-14 days have a weight of 0.7 and so on as illustrated in Table 1. [11]

Range	Last visit	Weight
1	0-4 days	1.0
2	5-14 days	0.7
3	15-31 days	0.5

4	32-90 days	0.3
5	91+ days	0.1
6	Never	0.0

Table 1: Example of frequency algorithm

The actual algorithm can be adjusted with the number of visits and some other points removed to keep the example simple. Figure 14 shows an example of frequent locations calculated by the back end.

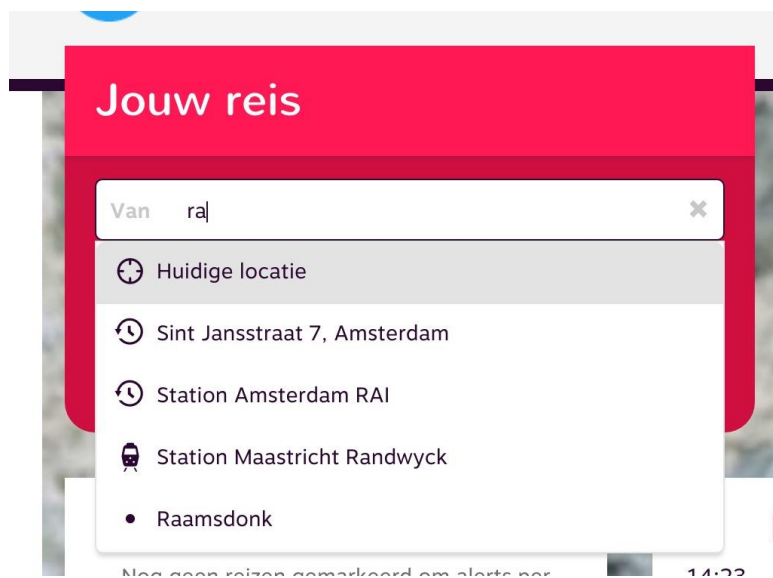


Figure 14: Example of frequent locations calculated by the back end

However, it is a good idea to keep the algorithm as simple as possible since the rating of frequent locations is calculated for each user personally and re-calculated after each trip so complicated calculations will give too much overhead for the back end services.

4.2 User's Preferred Types of Transport

Many users have personal preferences based on transport types. For example, the author prefers trains over buses because trains have free Wi-Fi allowing working during trips. However, the author's grandma prefers buses on the same routes because she

only has a bus discount. The author's best friend prefers bicycles over all the other types of transport because they are free and fast.

Every other person in a city also has his preferences. However, planners still ignore these preferences and always sort trip alternatives based on the fastest-first options.

The simplest version would be to remember the user's choice over time. The planner has to remember the most preferred types of transport which the user took in general and the types of transport usually taken in repeated routes (such as trips to an office and back).

One of the important points here is that 90% of the trips a user makes are repeated trips he commits every week. So the algorithm should put the biggest effort in optimizing these trips.

4.3 User's Email and Calendar

Taking the user's email and calendar into account is another possible option. If some meeting is marked in the calendar, it usually has a meeting point. The planner needs access to the user's calendar to be able to analyze it and to suggest a route right before the meeting. In other words, the app should send a push notification suggesting to travel to the meeting. And the suggestion should be based on how long the trip takes.

More complicated option is to also analyze user's emails and chats to find the possible meetings as shown in Figure 15.

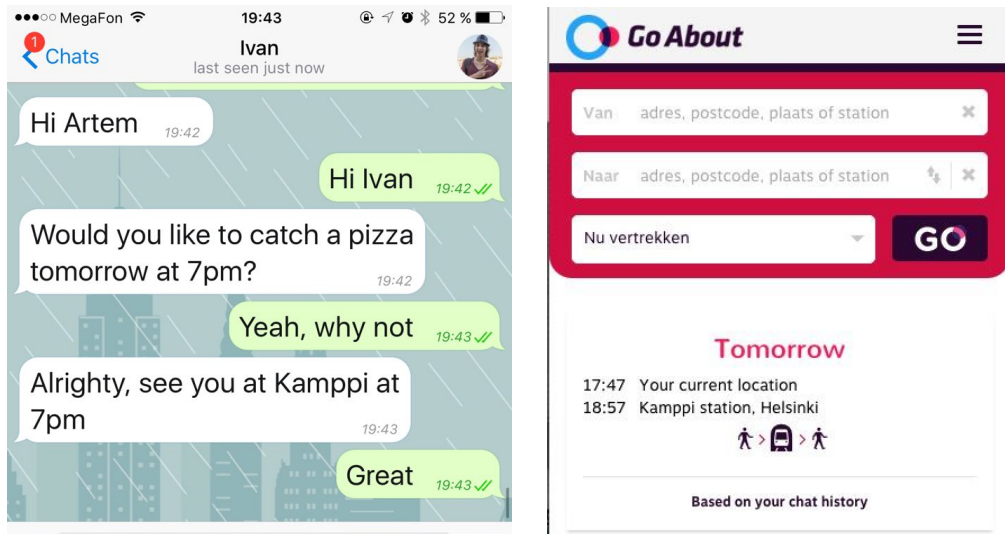


Figure 15: Planner suggests tomorrow's trip based on chat history

That requires the service to constantly parse user's accounts, analyze it and find matching patterns. On one side, there could be privacy issues, if hackers get access to planner's machines, they can also get access to all user's conversations. On the other side, seamless integration with calendar, email and chats allows to skip the planning part, all the discussed meeting are always automatically added to the journey planner.

4.4 Weather and Time the day

These are general personalization options, in most cases, these do not depend on a single user but are useful for inhabitants of the same region in general.

For example, statistically, in a warm good weather most people would prefer a walk over 1-2 bus stops, while in a cold or rainy weather, most users would take bus anyway. Same goes for the time of a day, during late evenings, the planner should suggest bus or taxi trip option over walks since it is safer.

4.5 Popular Destinations

During popular events and celebrations, it is easy to predict where most users would go. So the planner can already suggest a route to the celebration on day of the celebration. Figure 16 shows a planner suggesting a journey based on the most popular destinations today.

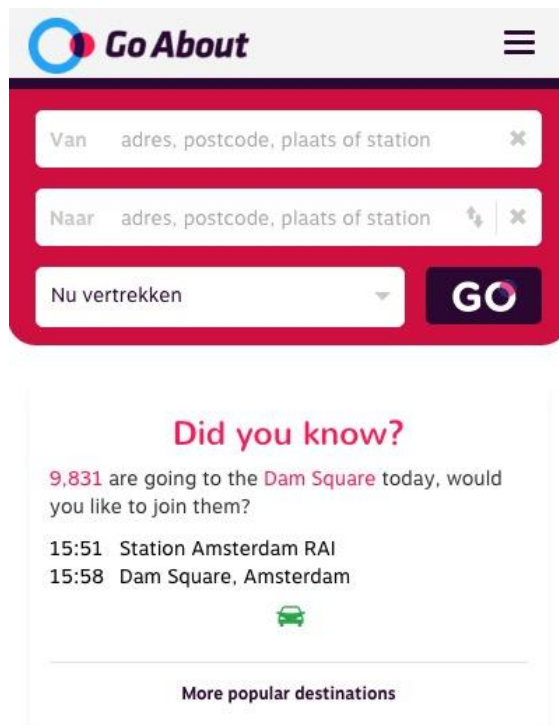


Figure 16: Planner suggests a journey based on the most popular destinations today

That option is nicely combined with analyzing the user's email for event tickets and other preferences.

Moreover, the planner does not need to have events typed by a human. It can suggest events automatically based on routes users planned in advance. For example, if 30,000 inhabitants planned a route to the main square tomorrow, it would be handy to suggest that route to other inhabitants of the area.

The planner can also work in the opposite direction. If too many people are planning routes to the same spot, the planner should give a notification about possible overcrowding and suggest a different route.

5 Structure of Prototype

GoAbout is a journey planner over the Netherlands which implements personalization options such as recent locations, saved trips, frequently used trips and remembering preferred transport types. It consists of OpenTripPlanner, back end and different front ends which can be Web-based front ends, mobile apps and nearly anything what can be of use route planners. For example, one of the front ends is a touch-based screen installed on Rotterdam central station as shown in Figure 17.



Figure 17: Example of an alternative front end. The picture is a property of the author

The prototype is open for the public and can be accessed via GoAbout website [5]

5.1 Open Trip Planner

Open Trip Planner (OTP) [6] is the most important part of GoAbout planner. However, it is not developed by the company, it is an open source project. It provides REST API for the back end and that is where the trips are actually planned. OTP takes departure and arrival coordinates and outputs up to dozen trip alternatives based on open data standards such as GTFS and OpenStreetMaps.

GoAbout project uses a modified version of OTP which also takes real time data into account and uses slightly modified algorithms for bike plannings. Bike is one of the most popular ways of transport in Holland and that fork of Open Trip Planner pays more attention to bike speed calculation, bike parking times and puts additional trip time when the bike is rented.

5.2 Back end

Back end is a layer of middleware between Open Trip Planner and front end. It transforms front end passed data to the format understandable for OTP and back. Also backed stores user data (and history) and all the additional services such as bookings, saved trips, predictions and so on. It is probably the most complicated part of GoAbout Infrastructure. However, that work gives a simplified overview of the back end because it is not the topic of the work but it is still a necessary part of the whole picture.

Back end is a resource-based API that uses the Hypertext Application Language (HAL) media format. HAL is a convenient format of hyperlinking between resources. Listing 5 shows an example of back end response in HAL-format.

```
{
  "type" : "stop",
  "label" : "Halte Weena, Rotterdam",
  "coordinates" : {
    "latitude" : 51.924016,
    "longititude" : 4.47568
  },
}
```

```

"name" : "Weena, Rotterdam",
"subtype" : "stopArea",
"_links" : {
  "http://rels.goabout.com/stoptimes" : {
    "href" : "https://api.goabout.com/location/stop/ad781278/
stoptimes{?interval}",
    "templated" : true
  },
  "self" : {
    "href" : "https://api.goabout.com/location/stop/ad781278"
  }
}
}

```

Listing 5: Example of back end response in HAL-format

Listing 5 is a typical back end response made in HAL-format. It demonstrates main benefits of HAL: The response is split into two parts — resource and links. Resource contains state (JSON data) and can also contain some embedded resources (like timetables, in our example). Links contain target (URI, Uniform Resource Identifier), relation (name of the link) and usually contain properties to be changed (for example, interval in stoptimes uri). [4]

The second big idea behind HAL is that links to different resources can change as long as their relation stays with the same name. Since all the front ends do not use straight links but use relations instead. Also that allows us to code generate almost the whole API level implemented on front ends.

```
location.get('http://rels.goabout.com/stoptimes', {interval: '3h'})
```

Listing 6: Example of front end request in HAL-format

Then stoptimes call on front end would look like code in Listing 6. Note that there are no straight links used, front end calls for stoptimes resource and defines link properties in the following object. The actual link and actual request are generated by the HAL-based realization.

Back end has over 30 different API calls and each of them might have some subcalls. However, that work gives a simplified overview of the back end because it is not the topic of the work but it is still a necessary part of the whole picture.

The whole description of back end API can be found in the official GoAbout documentation. [4]

5.2.1 Geocoder and Geodecoder

Geocoder is a service which converts location names into geo-locations with longitude and latitude coordinates. It allows to search for a location by a query. The query does not need to be an exact match or to be a start end of the string. Back end can handle mistakes in addresses or get the right results even in fairly complicated cases. For example, a query such as “ams ce” will return “Amsterdam Centraal” as the first result.

```
{
  "_links": {
    "http://rels.goabout.com/geocoder": {
      "templated": true,
      "link": "https://api.goabout.com/geocoder/
        encode {?query, count, types}"
    }
  }
}
```

Listing 7: A HAL resource representing geocoder API Method

“query” is a required property while “count” and “types” are optional. If type is specified, the back end will only return location of specific types, for example, address, city, stop, station, parking or something else.

```
{
  "locationHrefs" : [
    "https://api.goabout.com/location/stop/4b95d4b4",
  ],
  "_links" : {
```

```

    "self" : {
      "href" :
"https://api.goabout.com/geocoder/encode?query=amsterdam%20ce&count=3"
    }
  },
  "_embedded" : {
    "http://rels.goabout.com/location" : [ {
      "type" : "station",
      "label" : "Station Amsterdam Centraal",
      "coordinates" : {
        "latitude" : 52.3789197836,
        "longitude" : 4.90088939667
      },
      "code" : "asd",
      "name" : "Amsterdam Centraal",
      "_links" : {
        "http://rels.goabout.com/stoptimes" : {
          "href" :
"https://api.goabout.com/location/stop/4b95d4b4/stoptimes"
        },
        "self" : {
          "href" : "https://api.goabout.com/location/stop/4b95d4b4"
        }
      }
    }, {
      ...
    }
  ]
}

```

Listing 8: A HAL-based response on the geocoder request with a query='amsterdam ce' and count=3

Geocoder not only returns coordinates but also returns the type of the location, its full label, name, address, city and some additional links like link to timetables if it is a station or a stop.

Geodecoder is a service which does the opposite, it converts coordinates into a human-readable location, usually, taking the closest possible address as the expected location. For example, a query with the geo coordinates placed on the central station of

Amsterdam will return “Amsterdam Centraal” location which can later be used with plan API.

Geocoder is not a part of OTP, it is a service implemented by GoAbout and it is based on the information received in GTFS feeds and some other POI sources.

Journey planner uses geo coordinates for planning but users prefer addresses so that part of API gives the translations needed.

5.2.2 Trip Planner

Trip planner is the very main API method of the back end. It accepts place identifiers such as fromHref, toHref and either departure or arrival time and returns a list of trip alternatives.

fromHref and toHref are exact location URIs generated by the back end and received by geocoder/geodecoder calls. Arrival or departure are date-time values in ISO 8601 format. The request can have either arrival time or departure time but not both of them. Listing 9 gives an example of a plan request.

```
curl -X POST -H'Content-Type: application/json' -H' Authorization:
Bearer <authorization_token> -d '
{
  "fromHref": "https://api.goabout.com/location/station/asd"
  "toHref": "https://api.goabout.com/location/address/2b0392ab",
  "departure": "2013-08-13T13:30:00Z",
}' 'https://api.goabout.com/plan'
```

Listing 9: Example of a plan request in CURL form, formatted for better readability

That request is given in a form of CURL request to show the difference between the HAL-approach and old-fashioned CURL. As seen in the request, the user needs to manually specify content-type and authorization headers, search for the exact API link and request it. HAL-request would look like `api.post('http://rels.goabout.com/plan', { <plan_object> })`

The answer consists of a list of plan alternatives sorted by departure time.

The planner uses place identifiers instead of coordinates because of its internal structure. Some places or, better say, points of interest might change its coordinates while keeping the same ID. So back end does the job of taking ID and changing it to coordinates for OTP.

5.2.3 User-specific Data

Back end keeps some user data. Currently, that data consists of the user's email, the user owned transport (car, bicycle) and a history of planned trips.

The history of trips is carefully recorded after each user so it can be used for personalization options described above in the report. Currently, back end only returns frequent locations. The locations are calculated using same frequency algorithm as used in Mozilla Browser for calculating the most visited pages. The algorithm is already explained in section "Personalization and prediction options"

5.2.4 Reservations

Reservations are another step in making planning personal. Some routes allow taxi or rented bicycle modes so the app can automatically make the necessary reservations. In that case, the user needs to register and to fill in his personal details such as full name, email and phone number. Figure 18 illustrates a typical flow of making a reservation.



Figure 18. Typical flow of making a reservation

In most cases, reservations simply send an email to the company whose product is reserved as shown in Figure 18. However, companies can be attached to a more complicated API so GoAbout's reservations are integrated with the company's IT infrastructure.

5.3 Front end

As stated above, there could be many different front ends for a planner. It could be anything: a web application, a mobile application, a simple widget or even a publicly accessed touchscreen in the middle of some train station. Currently, there are: web-based application for both desktop and mobile users, iOS application for iOS users, a web-based front end for public touchscreens and iframe widget for websites allowing to plan journeys via it. Figure 19 shows the web front end for GoAbout Planner.

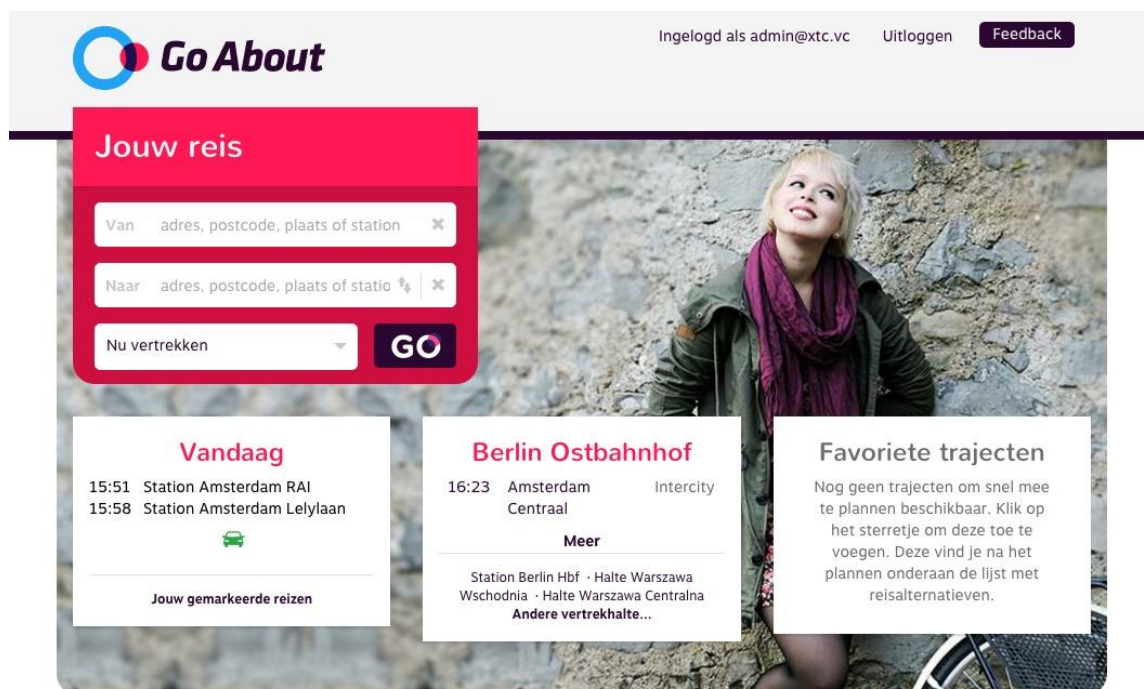


Figure 19: Web front end for GoAbout Planner

However, the most popular front end as for today is a web-based application. It can be opened on both desktop and mobile. One third of the users are mobile users.

It is the front end on which all the ideas are applied in the first place, including personalization and prediction options.

5.3.1 Structure Overview

The structure of the project is made with Angular Style Guide in mind. [16] It follows the basic LIFT principle where “L” stands for locating the needed file quickly, “I” for identifying the code at a glance, “F” for the flattest structure and “T” for trying to stay DRY (Don’t repeat yourself). The app is split into components, each component usually includes html view, controller and might have services, directives and filters. The structure is shown in Figure 20.



Figure 20: Structure overview, split into components

The front end is based on AngularJS as Javascript framework and Bootstrap as HTML & CSS framework. Altogether, that set allows developers to bring new features in hours.

AngularJS is a powerful engine for building modern applications. AngularJS allows to bind HTML, JS and CSS into one solid thing. Its greatest features include 2-way binding, e. g. all the JS variables are synchronized with HTML views. If the value is updated in either JS or HTML, it is automatically updated in all the other places tied to the value.

The HTML structure is based on Twitter's Bootstrap framework version 3.x. The general rule is to use Bootstrap components and standard CSS-classes whenever possible to reduce amount of custom code.

CSS is also based on Twitter's Bootstrap, however, all the default styles were overridden to fit GoAbout. The general idea is same as in the structure, do not repeat yourself. All the CSS classes are as generalized as possible so that they can be reused on different pages.

Also the project uses Stylus preprocessor to add variables, functions and mixing to CSS. It allows developers to quickly change one variable and change color of every element using that variable instead of changing each class individually.

5.3.2 User Interface Overview

5.3.2.1 Homepage

The user interface homepage consists of four main elements: Search form, already planned trips, closest train station and predictions module as illustrated in Figure 21.

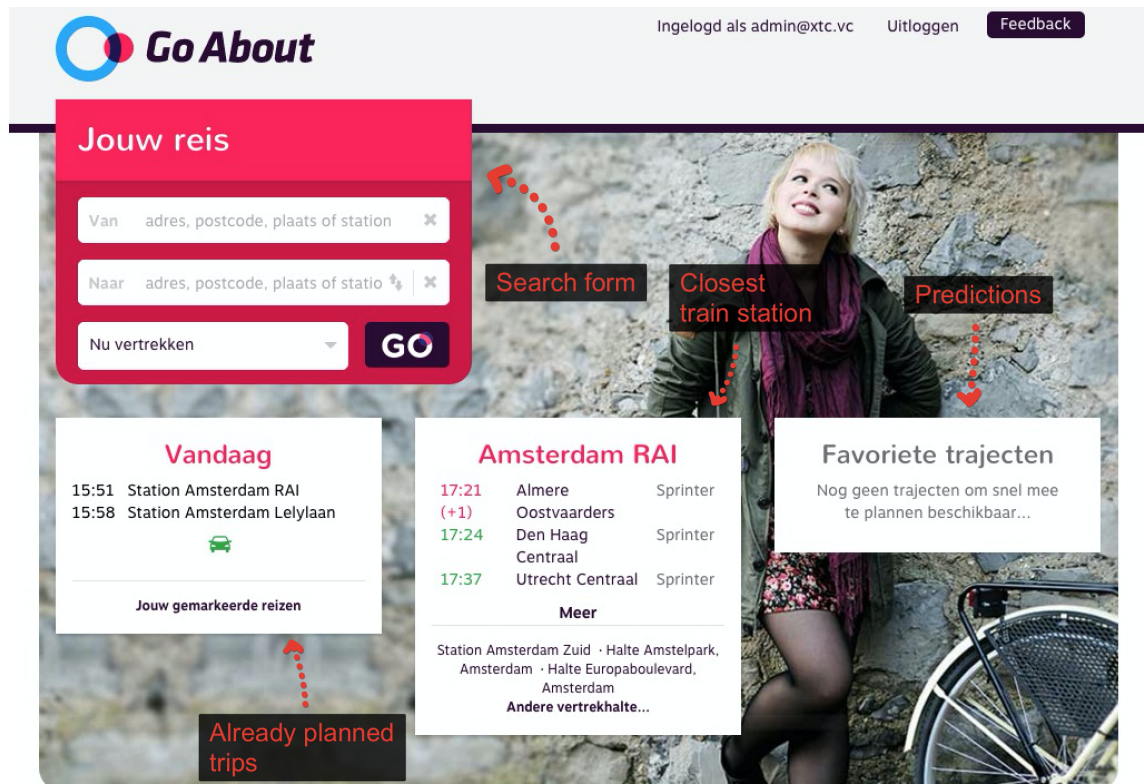


Figure 21: Homepage of GoAbout planner

The search form looks simple for the user and it has ordinary “from” and “to” fields. The label “Nu vertrekken” on the bottom means “Leaving now” and can be changed to a more exact time of “Arrival” or “Departure”. When “Go” button is pressed, the planner starts to search trip alternatives. Figure 22 shows an example of field autocomplete using frequent locations and fuzzy search

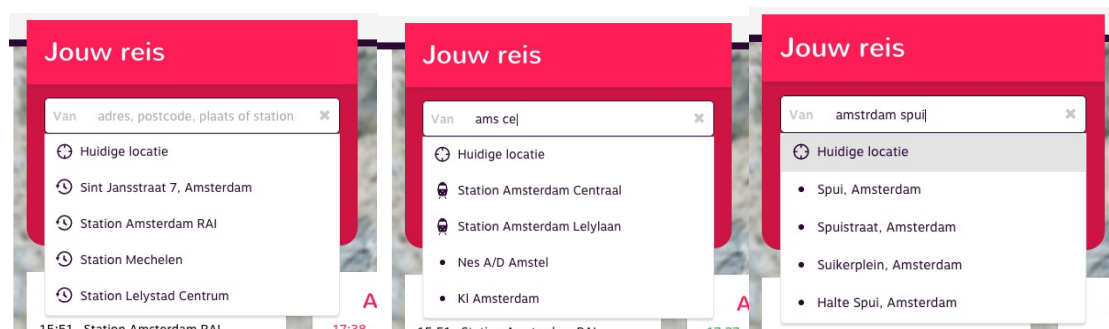


Figure 22: Example of field autocomplete using frequent locations and fuzzy search

However, each form field is not as simple as it looks. When there is nothing in the field, it shows “Current location” as the first option and the recently used locations as all the next options. As soon as something is typed in the form, the autocomplete algorithm

uses Fuzzy string searching. It is a technique used to find strings even if the match is not exact. As illustrated in Figure 22 the string “ams ce” was correctly determined as “Amsterdam Central” and the string “Amstrdam spui” was correctly determined as “Amsterdam Spui” even when it had a typo made by the user.

Although the algorithm is made on a database level to speed up the search, it needs additional explanations since it is a big step in form usability. That implementation of Fuzzy search is based on Levenshtein ideas firstly formulated in 1966. It generates all the possible matching terms that are within the maximum “error” rate distance specified. [12]

The “Already planned trips” block in Figure 21 is about saved routes, it shows all the journeys which are saved by the user by hand. It shows only the first three trips planned for today and the rest can be opened by clicking on the link on the bottom of the page.

The “stoptimes” block in Figure 21 is about the closest stops and stations. It shows the closest train station and the next few trains leaving from that station. In the future, that form will show the closest and the most frequently used by the user stop or station of any type.

The last block, “Predictions” in Figure 19, is experimental. It takes the most favorite user destinations and makes a quick planning to them. It works in a very simple way by taking the same frequent algorithm as described in “Personalization and Prediction Options” chapter. And then planning to the first 5 frequent locations (except the ones which are closer than 2 kilometers).

5.3.2.2 Planning and Routes

The planning consists of three parts: search form, list of trip alternatives and detailed description of each trip as illustrated in Figure 23.

Go About Inloggen Registreren Feedback

Planning form **Map**

Jouw reis

Van Station Amsterdam Centraal x

Naar Binkhorst 36, Rotterdam x

Nu vertrekken GO

Start reis-adviezen met: alle

Eindig reis-adviezen met: alle

13:25 - 14:33
Intercity direct vanaf Amsterdam Centraal ... 01:08

12:55 - 14:03
Intercity direct vanaf Amsterdam Centraal ... 01:08

12:55 - 14:13
Intercity direct vanaf Amsterdam Centraal ... 01:18

12:55 - 14:18
Intercity direct vanaf Amsterdam Centraal ... 01:23

13:10 - 14:25
Intercity direct vanaf Amsterdam Centraal ... 01:15

13:10 - 14:26

Station Amsterdam Centraal → Binkhorst 36, Rotterdam

Filtering

Het is momenteel niet mogelijk om een actuele planning te geven.

01:08

Intercity direct Richting Breda
00:41 12:55 Station Amsterdam Centraal, spoor 15b
13:36 Station Rotterdam Centraal, spoor 3

Onbeperkt reizen in Amsterdam? Koop een GVB dagkaart [Google](#)

Metro D Richting De Akkers
 00:41 13:37 Halte Rotterdam Centraal, spoor 3
13:48 Halte Zuidplein, Rotterdam

Trip alternatives **Detailed Description**

Figure 23: Route page

The search form has been already described in “Home page” section, it the same form. However, it has one more application here. It lets users correct their departure or destination locations right on the page without going back saving a few moments of time.

The trip alternatives section is a short list of different routes. Usually, there are 3-10 different options and as long as the prediction algorithms are not perfect, a bit wider choice of options is preferred. Each option shows icons of transport types used (and hence number of transfers), departure and arrival times, length of the trip and a short label stating the most relevant info. In case of train trips, it will be the heading direction, in case of car or bicycle — distance between locations. All the information allows users to quickly browse through the list without clicking each option in deep.

If the user is consistent in his choices, he can set a “start with” or “end with” filter. Then the app will hide all the options which do not fit the criteria. The filter remembers the last chosen options so all the trips made afterwards will be filtered in the same way, except when there are no trips fitting the criteria, then the filter will be reset to default options.

Next, the most important part of the planner is the detailed trip description. That part takes 50% of the trip page and it consists of the map on the top, description of each trip part in the middle and a few small notes including price of the trip, total duration and the amount of CO₂ gas extracted during the trip on the bottom. The last point allows to easily feel the difference between a bike trip and a car trip on the same distances.

The map is based on OpenStreetMap and it allows users to move it or zoom it. Also clicking on each transfer will zoom the map on that transfer.

The detailed transfer description includes the type of the transport used, line number or head sign label (if needed), exact times of boarding/deboarding from the transfer and labels of departure arrival station as illustrated in Figure 24.

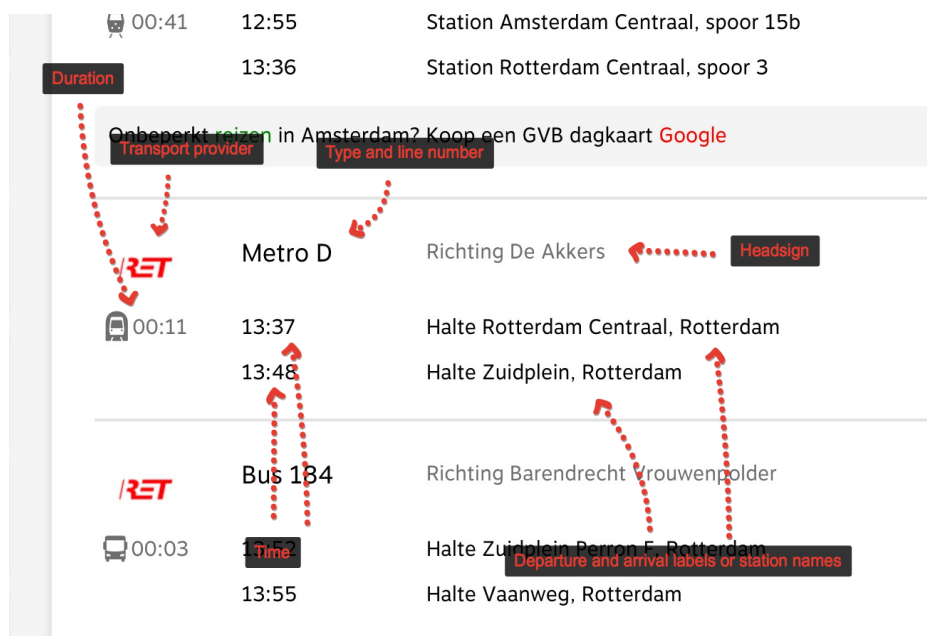


Figure 24: Each transfer in deep

The page is optimized for mobile devices and it can be shared both via simply copy-pasting the link (then the planner will calculate the same trip again) and clicking the email button, when the planner will send an email with exactly that trip option.

5.3.2.3 User Profile

The user profile and subscriptions allow the planner to give more personalized routes. The general profile page as illustrated in Figure 25 allows the user to choose his preferred bicycling speed and his home location. With the home location the planner can count on the user's bicycle. It will only give the bicycle planning options if home is closer than 15 kilometers. If home is further than that, the planner might still show bicycle options but will also suggest the nearest bike rentals.

Figure 25 shows the...

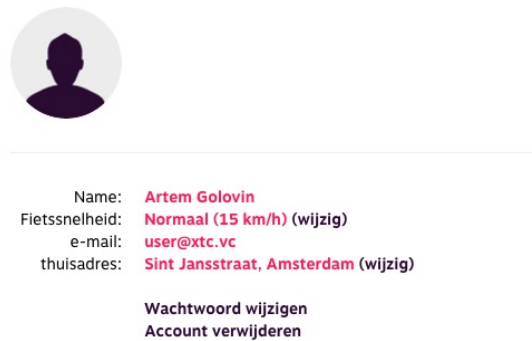



Figure 25: General profile page

The subscription page illustrated in Figure 26 allows the user to mark what types of transport he owns or is able to rent/use. For example, it allows to explicitly mark that the user owns a car, a bicycle and a few additional bicycles on different stations. The last point might not make any sense for international users but many Dutch people own a few bicycles located on different stations so they can use it when arriving to that specific station.

Jouw reisopties

Hieronder staan de opties waar jij mee kunt reizen.

	Jouw auto ✕
	Jouw fiets ✕ De thuisfiets is beschikbaar in een omtrek van 15km. vanaf je thuisadres .
	Stationsfiets ✕ Locatie Station Amsterdam Centraal

Voeg reisopties toe aan jouw vloot.

	OV-Fiets OV-Fiets is een makkelijke huurfiets voor het laatste stuk van je reis. Voeg toe
	Regiotaxi Noordoost-Brabant Vervoer met de Regiotaxi Noordoost-Brabant. Voeg toe
	Munckhof Taxidienst Munckhof is al jaren gespecialiseerd in het regisseren en verzorgen van personenvervoer en reizen voor overheden en voor de zorgsector. Voeg toe
	Regiotaxi Midden-Brabant (0800 483 17 62) De provincie Noord-Brabant en 8 samenwerkende gemeenten in Midden-Brabant hebben het Servicepunt georganiseerd om gebruikers te informeren en nazorg te bieden. De gemeenten zijn: Dongen, Gilze en Rijen, Goirle, Hilvarenbeek, Loon op Zand, Oisterwijk, Tilburg en Waalwijk. Samen zijn zij partners in Regiotaxi Midden-Brabant. Bel voor een reservering van de Regiotaxi Midden-Brabant naar 0800 483 17 62 Voeg toe

Figure 26: Subscriptions on the profile page

Also subscriptions might include some services, for example, taxi subscription or rental bicycle subscription. If the subscriptions are already present, trips using these subscriptions gain additional weight in the trip alternatives so they will be shown whenever possible.

5.3.2.4 Transport Providers and Reservations

Transport providers are extensions to the planner in some way. Some transport providers can add additional trip options. For example, users with a taxi subscription will get taxi options with a “reserve” button as illustrated in Figure 27. Since users already have subscription details typed in their profile and the route planned, the booking form will be filled for them. The only thing the user needs to do — is to hit the “confirm” button.

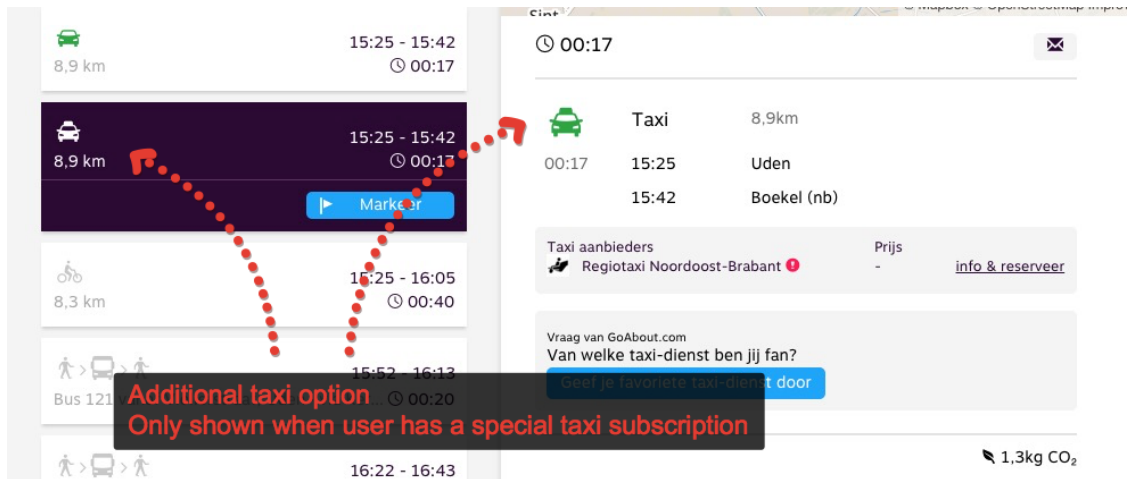


Figure 27: special trip alternative shown only to users with subscription

Another use of the module is small proposals of different reservations. For example, the planner can propose some bicycle rentals or a car rental if the trip consists of long walks as illustrated in Figure 28.

	Lopen	0,5km	
00:07	13:56	Halte Vaanweg, Rotterdam	
	14:03	Binkhorst 36, Rotterdam	
Huurauto's			
	Regina (0,3km)	Prijs	€3,- per dag info reserveer
	jacob (0,7km)		€3,- per dag info reserveer
	Leon (0,8km)		€3,- per dag info reserveer
	Hermans V40 op LPG (1,8km)		€23,- per dag info reserveer
	Adriana's Wheels (1,8km)		€25,- per dag info reserveer
Huurfietsen			
	Rotterdam Zuidplein (1,1km)	Prijs	€3,- per dag info
	Rotterdam Lombardijen (2km)		€3,- per dag info

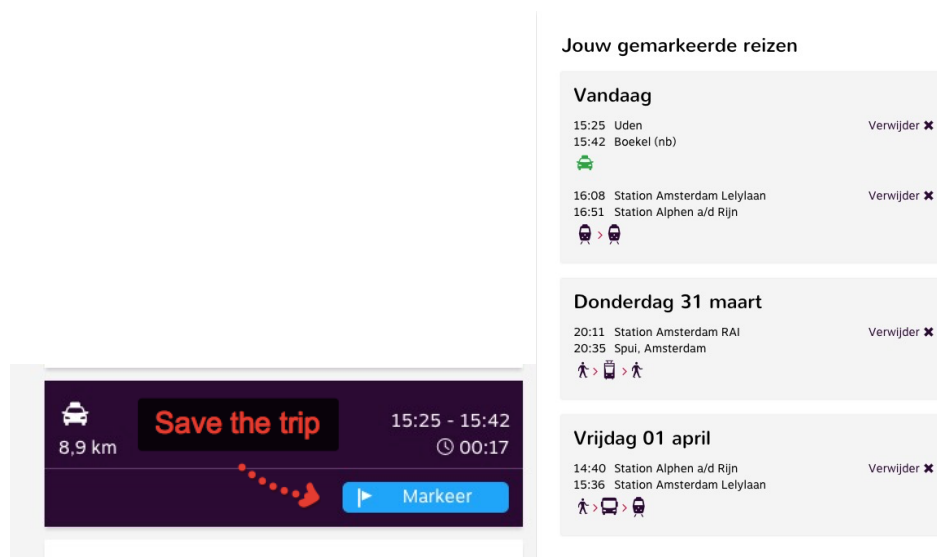
Figure 28: Proposals of car or bicycle providers nearby

These reservations can be anything, however the most popular options are parking lots and bicycle rental. Each proposal includes an approximate price and a button to either get additional info or to reserve the option right on the planner's website.

All these options allow users to plan everyday journeys and make booking for them without leaving the website.

5.3.2.5 Saved Trips

The saved trips functionality is used to manually save a specific trip for the future use. It is useful to make plannings for the future so users do not need to plan again, the saved trip can be reused. The functionality consists of two parts: the “Save trip” button as shown in Figure 29 and an overview of already saved trips.



Figures 29 and 30: “Save trip” button, displayed after each trip alternative and a list of saved trips

The closest three saved trips are displayed on the front page so these are always visible. The list with all the trips can be checked on a separate page called “Saved trips”. The page is shown in Figure 30. All the “past” trips are automatically removed and the user can manually remove future trips if needed.

5.4 Summary

GoAbout planner already implements the following five steps in personalization:

- Location predictions in the search form where the user can plan from his current destination to the destination which was recently or frequently used. However, it uses a simple frequency algorithm. Predictions can be improved by using some of machine learning.
- Journey predictions based on the most frequently used routes. That section can also be improved by remembering the times when the trip usually takes place.
- Remembering filter options. That is a simple form of remembering favorite transport types. Once the user sets his preferred transport, the app will filter all the excluded types of transport.
- Adding own types of transport and adding subscriptions. Users can mark if they have or do not have car/bicycle. In that case, trips and pricing will be calculated with using own transport or rented one. Also user can add a subscription, for example, local taxi subscription. And then the planner will calculate the route using that taxi service as the preferred.
- Manually saved trips and sharing. The planner allows users to save planned routes for the future so trips can be planned in a few days in advance and these trips can be later accessed from any device, including mobile.

GoAbout still investigates the positive impact of these improvements so no exact statistics is yet available. However, most of the feedback states that these steps already make a difference, save a few seconds every time when the planner is used. However, a good product is all about small details. The more of it are added, the more practical the planner gets.

6 Discussion and Conclusion

The current planning systems are not ideal and they are too generic. For example, the current version of CityMapper or GoAbout shows up to 12 different trip options. So many planners show way too many options to the user in every possible aspect. There is no

need to show a car for users who do not have one or to show a bicycle route for 50km long trips.

For example, based on the research made in GoAbout, there is no need to show a search field in the first place. 95% of the trips are made to previously used locations so a list of the frequent (recent and frequent) locations is more useful. There is even no need to spend API resources to search in all the locations in the country, in most cases the address required by the user is no more than 20km away.

The illustrated planner already implements some of the described ideas. It remembers the last and frequent searches and shows them in the planner page. Also it goes even further and proposes plans to the most frequently used destinations. More advanced systems should also count on the user's type of transport in each specific case (for example, if the user uses a car for shopping and public transport for going to work, the system should remember that), count on the weather (biking in rain is not the nicest thing to do) and on the calendar's history (if there is an appointment it is possible, that it requires travelling) or email analysis (gmail already highlights flights and that is only a step away from highlighting closer destinations). Plus other improvements as described in Section 6 "Personalization and Prediction Options".

The demo only takes the user's history as input. But the next big step in the planning systems is to record every user move, analyze it and predict future user movements based on many factors.

As soon as planning systems learn to analyze every person personally, they would be able to generate common travel destinations for neighbors of the same house or of the same district. Planning systems would be able to dynamically change public routes to fit the needs in the exact same period of time since it is no secret that some routes are crowded in the morning but completely different routes are crowded in the evening. Also planning systems would be able to instantly detect that if someone is going in the same direction, then that person could share a ride so that would lower the number of cars on the road and hence make traffic jams smaller. If there are too many people going in the same direction, planning systems should be smart enough to propose a different place (for example, different supermarket) or just ask to wait for a while.

All the automatization already lowered the amount of traffic jams in major cities and reduced travel times for many inhabitants. So the smarter systems get, the better they would be able to predict our movement and the better they would change our or someone else's routes to make everyone's journeys faster.

At some point, planner systems may get so smart they would become extinct in a way we know them today. An ideal interface is the interface which does not exist. So an ideal planning system might be the one which will always know where we are going so it will automatically and in time get us a cheap shared taxi which will get us to the destination so that we will not even need to open the app.

References

1. **Ger Haartman, Wim de Bell and Wouter Overhaus.** Mapping History — digital historical atlas. [Online] No date. [Cited: 22 February 2016] <http://www.mappinghistory.nl/>
2. **Transport for London.** London underground, london buses. [Online] No date. [Cited: 22 February 2016] <https://tfl.gov.uk/corporate/about-tfl/culture-and-heritage/londons-transport-a-history>
3. **Jeremy Norman.** The First Operational Satellite Navigation System. [Online] No date. [Cited: 22 February 2016] <http://www.historyofinformation.com/expanded.php?id=106>
4. **Mike Kelly.** HAL - Hypertext Application Language. [Online] 13 June 2011. [Cited: 29 February 2016] http://stateless.co/hal_specification.html
5. **GoAbout.** Official GoAbout API Documentation. [Online] 03 May 2011. [Cited: 29 February 2016] <https://apidocs.goabout.com>
6. **OpenTripPlanner.** Official documentation. [Online] 10 February 2016. [Cited: 11 March 2016] <http://docs.opentripplanner.org/en/latest/>
7. **Google Developers.** General Transit Feed Specification. [Online] 27 May 2015. [Cited 12 March 2016] <https://developers.google.com/transit/gtfs/>
8. **Google Developers.** General Transit Feed Specification Real-time. [Online] 29 July 2015 [Cited 12 March 2016] <https://developers.google.com/transit/gtfs-realtime/>
9. **Sergey Apelsberg,** OpenStreetMap as the source geodatabase. [Online] 17 November 2015 [Cited 12 March 2016] <https://habrahabr.ru/post/270513/>
10. **J. Koszelew,** The Theoretical Framework of Optimization of Public Transport Travel, 6th International Conference on Computer Information Systems and Industrial Management Applications IEEEICISIM'07, pp 65-70, 2007.
11. **Mike Connor,** Places Frecency [Online] 19 November 2010, [Cited 16 March 2016] <https://wiki.mozilla.org/User:Mconnor/PlacesFrecency>
12. **Esko Ukkonen,** Algorithms for approximate string matching, Information and Control. Academic Press, 1985, volume 64, p. 100-118
13. **Ivan Sychev,** History of Navigators. [Online] 11 March 2014, [Cited 6 April 2016] <https://habrahabr.ru/company/boxowerview/blog/214333/>
14. **Zhonghua Wang,** A Preliminary Study of the Global Distribution Systems, IEEE, 2010, p. 60-63, DOI: 10.1109/ICIME.2010.5478307

15. No author, Pääkaupunkiseudun matkailijoille uusi reittiopas netissä. [Online] 14 November 2001, [Cited 06 April 2016] <http://archive.is/kOGQ>
16. John Papa, Angular Style guide. [Online] 27 July 2014, [Cited 13 April 2016] <https://github.com/johnpapa/angular-styleguide>
17. Yandex, How Yandex.Traffic works [Online] No Date available, [Cited 28 April 2016] <https://yandex.ru/company/technologies/yaprobki/>