# Open Source Solution. Technical documentation

OpenSource

Exported on 11/04/2021

# Table of Contents

1 Open Source overview ..... 5
1.1 System purpose ..... 6
1.2 Brief description ..... 6
1.3 Access to Open Source code and test servers ..... 7
1.4 Technical support service ..... 7
1.5 License agreement ..... 7
2 Open Source system architecture ..... 17
2.1 General system architecture ..... 17
2.2 Basic variables types and descriptions ..... 20
3 Server solutions ..... 22
3.1 Network infrastructure ..... 22
3.1.1 Infrastructure deployment ..... 22
3.1.1.1 Kubernetes cluster ..... 22
3.1.1.2 GIS Host ..... 23
3.1.2 OSRM Map update ..... 27
3.1.2.1 Preparation ..... 27
3.1.2.2 OSRM ..... 27
3.1.2.3 Nominatim ..... 28
3.2 Internal services ..... 29
3.2.1 Client-server data exchange ..... 29
3.2.2 Scoring Proxy - authentication ..... 32
3.2.3 DeviceInfo ..... 33
3.2.4 DeviceInfo (SQL) ..... 33
3.2.5 Mobile Calculator ..... 33
3.2.6 Mobile Events Database ..... 34
3.2.7 Mobile receiver ..... 35
3.2.8 Scoring API - Mobile ..... 36
3.2.9 Scoring (SQL) ..... 36
3.2.10 Scoring Job ..... 38
3.2.11 Scoring Queue ..... 38
3.2.12 Trip job ..... 38

3.2.13 Trip triggers database ........ 38
3.2.14 Trips Queue ........ 39
3.2.15 LinkMapper ........ 39
3.2.16 Nominatim ........ 40
3.2.17 RoadInfo Service ........ 40
3.2.18 OSRM ........ 40
3.3 External services ........ 41
3.3.1 Google StackDriver ........ 41
3.3.2 Open Street Map API ........ 47
4 Mobile solutions ........ 48
4.1 Android ........ 48
4.1.1 Get started Android ........ 48
4.1.2 Initialization Android SDK ........ 50
4.1.3 Onboarding & main screens for Android ........ 53
4.1.4 Sample App. Android ........ 64
4.2 iOS ........ 67
4.2.1 Get started iOS ........ 67
4.2.2 Initialization iOS SDK ........ 68
4.2.3 Onboarding & main screens for iOS ........ 73
4.2.4 Sample App. iOS ........ 94
4.3 Server interaction methods ........ 103
4.3.1 Obtaining trips data ........ 103
4.3.2 Obtaining data on the details of the trip ........ 110
4.3.3 Obtaining the average score and recorded mileage ........ 116
4.4 SDK performance methods ........ 121
4.4.1 Start and stop of trip detection ........ 121
4.4.2 Penalties and scores ........ 125
4.4.2.1 Penalty types ........ 126
4.4.2.2 Mobile penalty calculation ........ 127
4.4.2.3 Server penalty calculation ........ 130
4.4.3 UI localization ........ 133

# 1 Open Source overview

**Kasko2go Open source solution in Usage-Based Insurance**

Many Insurance Companies have LOSS RATIO of 80% which they want to reduce.

The kasko2go Open Source solution in the field of motor insurance allows to evaluate the individuality of every driver, geography of his/her mobility, driving behaviour, mileage and a number of other parameters. Understanding personal risks of a certain driver allows insurance companies to develop insurance products for road transport with fine tuning of risks for every policyholder.

The Open Source solution gives a driver and an insurance company an opportunity for constant monitoring events related to vehicle movement beginning with the motion start and up to vehicle stop. The kasko2go Open Source solution includes three elements:

- a mobile application to install on the driver's smartphone with Android or iOS;
- a server platform which collects and processed data received from the mobile application;
- Scoring monitor service, which is designed to view information about drivers' trips.

Tasks of the mobile application:

- authorization of a user and the user's mobile device;
- setting up access to smartphone sensors;
- building safe routes for vehicle motion;
- determining the start and finish of vehicle motion;
- collection and transferring trip telemetry data (GPS coordinates, smartphone accelerometer data) to the server platform;
- display of trip history.

The server platform of the kasko2go Open Source solution is built on the basis of Google Cloud services and performs the following tasks:

- receiving telemetry data from the mobile application;
- matching GPS coordinates of vehicle movement to the road map;
- trip routes recording;
- determining safety objects on the vehicle route, near which specific behaviour rules should be observed, for example, speed limiting;
- calculating a score of user's driving quality;
- determining of driver actions that pose a potential hazard to road traffic.

The key element of the kasko2go Open Source solution is the unique scoring model developed by our specialists. Our scoring model is based on unique algorithms for data analysis developed on the basis of a tremendous amount of statistical information received from vehicle OBD controllers.

The scoring model developed by our company allows:

- determining potentially hazardous events in the user's driving behaviour;
- calculating:
    - duration and length of each trip;
    - score of each specified trip;
    - weighted average trip score for a specific period of time;
    - accident rate.

Today our software products built on the basis of kasko2go Open Source solution already have practical implementations that provide:

- car policyholder with:
  - transparency in forming the insurance policy cost;
  - personal control over tariffs;
  - reduction of insurance tariffs (up to 50% in comparison with traditional offers), in particular, for young drivers and drivers with a small annual mileage;
  - flexible system of sharing a car with family members and friends;
  - increasing the level of traffic safety in general: improving driving behaviour quality, reducing the frequency and the degree of severity of accident consequences;
- insurance companies with:
  - better risk management;
  - client base growth due to attracting low-risk clients through individualised pay-as-you-drive offers;
  - improving communication with clients and, as a result, increasing the level of their satisfaction, loyalty and client base retention.

We offer to try our kasko2go Open Source solution. For this, we provide an access to the server infrastructure, sample application for Android and iOS operating systems, and SDK that we have developed.

## 1.1 System purpose

**Project Title:** Intelligent Information System for Developing and Analysing Vehicle Driver Profile

**Project Objective:**

- Reduction of payouts due to minimising risks of causing accidents by a driver (the insured) through estimating the probability of a driver being involved in an accident based on the analysis of driving behaviour, routes and driving conditions.
- Increasing company's profits based on validated correction of insurance terms and conditions and reduction of payouts on claims taking into account a predictive risk evaluation of causing accidents by a driver (the insured) through estimating such probability using an automated analysis of driving behaviour, routes and driving conditions.

**Application Functionality:**

- Collection and primary processing of telemetry data on vehicle movement taking into account the environmental conditions.
- Evaluation of risk (probability) of causing an accident by a driver.
- Development of recommendations (for a driver, an insurance company manager).
- Predictive evaluation of an insurance company savings (or costs) for different variants of customer insurance terms and conditions.

## 1.2 Brief description

Every car driver has an own, unique driving behaviour, usual routes, mileage during a certain period of time. The driving behaviour and routes can depend on a number of parameters such as a season, time of day, weather conditions, a road type. All these parameters eventually determine the mileage and allow determining the accident rate regarding the conditions, in which a driver drives a vehicle, and in relation to other drivers.

The service is developed to determine a driving behaviour of a certain driver. Through this service companies can 'encourage' drivers to drive in a more moderate or calm manner, which allows increasing road safety. For example, for the drivers practicing 'safe' driving behaviour:

- insurance companies can reduce insurance premiums;

- car rental is cheaper (for example, carsharing);
- vehicle fleet owners can 'punish' dangerous drivers up to firing them;
- banks can offer different loan terms and so on.

The Open source solution service works with digitised road maps, has the information on static and dynamic (current) accident rate of different road sections. The accident rate, in turn, depends on the time of day, traffic density (congestions), weather conditions and dozens of other parameters.

Using IoT devices for a car, mobile applications as well as data from partner companies, we have collected statistical data on mileage on trillions of kilometres of roads, millions of trips and thousands of accidents. The analysis of this data allowed us to develop a model for calculating driving behaviour of a driver, which is maximally close to the actual driving behaviour. Once all necessary calculations have been performed, a decision on ways of encouraging or punishing a driver (the insured) for the driving behaviour is taken depending on the results obtained.

The Open source solution service is developed according to the ‘client-sever’ architecture and includes:

- server software;
- client software:
  - sample app for Android;
  - sample app for iOS.

See the Technical requirements(see page 17) section to check out the parameters of the server infrastructure which maintain the system performance. An example of deploying the server infrastructure is described in the Infrastructure deployment(see page 22) section.

A driver’s smartphone is used as a client device. An accelerometer and a gyroscope, modules of GPS, GSM and Bluetooth systems embedded into a smartphone turn a driver’s smartphone into a source of telematic information. A mobile application installed on the smartphone transmits telematic data to servers for further processing. See the Technical requirements(see page 17) section to check out the smartphone requirements.

## 1.3 Access to Open Source code and test servers

Access to the system source codes and test servers is provided on the terms set forth in the annex to the Agreement.

## 1.4 Technical support service

An Open source solution Technical support center is an institution that provides day-to-day operational support and services to customers.

You can contact us: info@kasko2go.com[1]

## 1.5 License agreement

**GNU GENERAL PUBLIC LICENSE**[2]

Version 3, 29 June 2007

Copyright © 2007 Free Software Foundation[3], Inc.

1 mailto:info@kasko2go.com
2 https://www.gnu.org/licenses/gpl-3.0.ru.html
3 https://www.fsf.org

**Preamble**

The GNU General Public License is a free, copyleft license for software and other kinds of works.

The licenses for most software and other practical works are designed to take away your freedom to share and change the works. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change all versions of a program--to make sure it remains free software for all its users. We, the Free Software Foundation, use the GNU General Public License for most of our software; it applies also to any other work released this way by its authors. You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for them if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs, and that you know you can do these things.

To protect your rights, we need to prevent others from denying you these rights or asking you to surrender the rights. Therefore, you have certain responsibilities if you distribute copies of the software, or if you modify it: responsibilities to respect the freedom of others.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must pass on to the recipients the same freedoms that you received. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

Developers that use the GNU GPL protect your rights with two steps: (1) assert copyright on the software, and (2) offer you this License giving you legal permission to copy, distribute and/or modify it.

For the developers' and authors' protection, the GPL clearly explains that there is no warranty for this free software. For both users' and authors' sake, the GPL requires that modified versions be marked as changed, so that their problems will not be attributed erroneously to authors of previous versions.

Some devices are designed to deny users access to install or run modified versions of the software inside them, although the manufacturer can do so. This is fundamentally incompatible with the aim of protecting users' freedom to change the software. The systematic pattern of such abuse occurs in the area of products for individuals to use, which is precisely where it is most unacceptable. Therefore, we have designed this version of the GPL to prohibit the practice for those products. If such problems arise substantially in other domains, we stand ready to extend this provision to those domains in future versions of the GPL, as needed to protect the freedom of users.

Finally, every program is threatened constantly by software patents. States should not allow patents to restrict development and use of software on general-purpose computers, but in those that do, we wish to avoid the special danger that patents applied to a free program could make it effectively proprietary. To prevent this, the GPL assures that patents cannot be used to render the program non-free.

The precise terms and conditions for copying, distribution and modification follow.

**TERMS AND CONDITIONS**

0. Definitions.

“This License” refers to version 3 of the GNU General Public License.

“Copyright” also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

“The Program” refers to any copyrightable work licensed under this License. Each licensee is addressed as “you”. “Licensees” and “recipients” may be individuals or organizations.

To “modify” a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a “modified version” of the earlier work or a work “based on” the earlier work.

A “covered work” means either the unmodified Program or a work based on the Program.

To “propagate” a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To “convey” a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays “Appropriate Legal Notices” to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

1. Source Code.

The “source code” for a work means the preferred form of the work for making modifications to it. “Object code” means any non-source form of a work.

A “Standard Interface” means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The “System Libraries” of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A “Major Component”, in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The “Corresponding Source” for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work's System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

2. Basic Permissions.

All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

- a) The work must carry prominent notices stating that you modified it, and giving a relevant date.
- b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to “keep intact all notices”.
- c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.
- d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an “aggregate” if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

- a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.
- b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or

customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.
- c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.
- d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.
- e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A “User Product” is either (1) a “consumer product”, which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, “normally used” refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

“Installation Information” for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

7. Additional Terms.

“Additional permissions” are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

- a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or
- b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or
- c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or
- d) Limiting the use for publicity purposes of names of licensors or authors of the material; or
- e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or
- f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors.

All other non-permissive additional terms are considered “further restrictions” within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An “entity transaction” is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party's predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

11. Patents.

A “contributor” is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor's “contributor version”.

A contributor's “essential patent claims” are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, “control” includes the right to grant patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor's essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a “patent license” is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To “grant” such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. “Knowingly relying” means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient's use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is “discriminatory” if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

12. No Surrender of Others' Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License “or any later version” applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

**How to Apply These Terms to Your New Programs**

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively state the exclusion of warranty; and each file should have at least the “copyright” line and a pointer to where the full notice is found.

```
<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year>  <name of author>

This program is free software: you can redistribute it and/or modify
it under the terms of the GNU General Public License as published by
the Free Software Foundation, either version 3 of the License, or
(at your option) any later version.

This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
GNU General Public License for more details.

You should have received a copy of the GNU General Public License
along with this program.  If not, see <https://www.gnu.org/licenses/>.
```

Also add information on how to contact you by electronic and paper mail.

If the program does terminal interaction, make it output a short notice like this when it starts in an interactive mode:

```
<program>  Copyright (C) <year>  <name of author>
This program comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it
under certain conditions; type `show c' for details.
```

The hypothetical commands `show w' and `show c' should show the appropriate parts of the General Public License. Of course, your program's commands might be different; for a GUI interface, you would use an “about box”.

You should also get your employer (if you work as a programmer) or school, if any, to sign a “copyright disclaimer” for the program, if necessary. For more information on this, and how to apply and follow the GNU GPL, see <https://www.gnu.org/licenses/ >.

The GNU General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License. But first, please read <https://www.gnu.org/licenses/why-not-lgpl.html >.

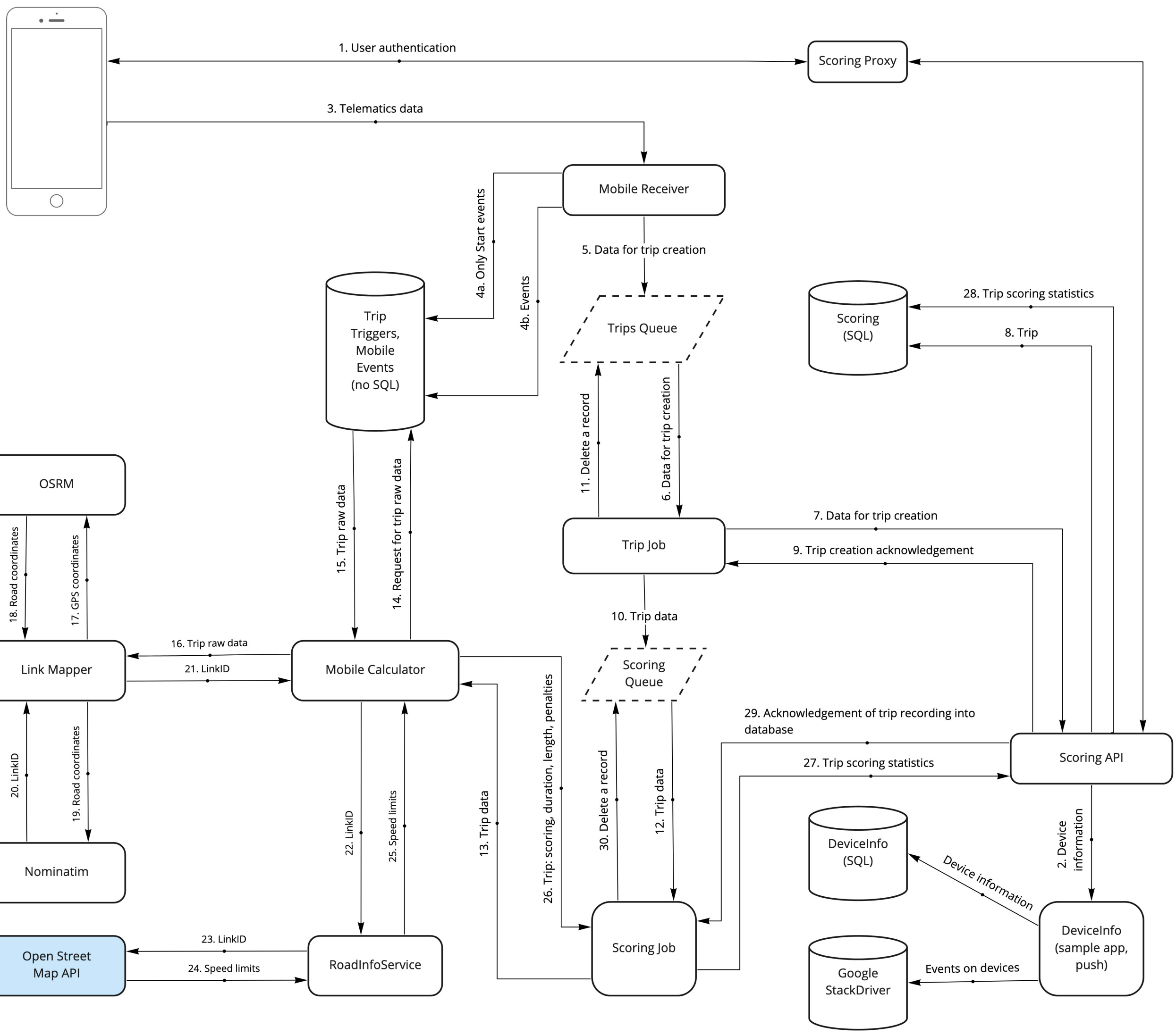
# 2 Open Source system architecture

## 2.1 General system architecture

The Open source solution is built according to the “client-server“ architecture and includes the following microservices.

| Microservice | Purpose |
|---|---|
| DeviceInfo (sample app, push)(see page 33) | Stores information about mobile devices of users (DeviceInfo (SQL)) and about events related to users devices (Google StackDriver). |
| Mobile calculator(see page 33) | Interacts with external and internal services, enriches raw telemetry data, calculates a trip score. |
| MobileEvents(see page 34) | Stores data on all events coming from a mobile device. |
| Mobile receiver (see page 35) | Provides control over the connection between the mobile application and the server, obtaining telemetry data from a mobile device. |
| Scoring API (see page 36) | Provides databases interaction. |
| Scoring Database(see page 36) | Stores a list of trips, their corresponding values of scores and list of violations that took place while driving. |
| Scoring Job(see page 38) | Provides interaction between queue processing services, databases and data processing services. |
| Scoring Proxy(see page 32) | Responsible for mobile user authentication. Mobile application provides authentication of a user and a user’s smartphone. |
| Scoring Queue(see page 38) | Trips queue to be enriched with data and score calculation. |
| Trip job(see page 38) | Forms trip data sets with the frequency of 1 Hz. |
| Trip triggers(see page 38) | Stores data on events that identify a trip start. |
| Trips Queue(see page 39) | Data queue for trip formation. |
| Link Mapper(see page 39) | Responsible for mapping GPS coordinates to the road identifier (LinkID). |

| Microservice | Purpose |
|---|---|
| OSRM(see page 40) | Matches GPS coordinates to the coordinates of the nearest road. |
| Nominatim(see page 40) | Matches the road identifier LinkID to a group of coordinates on the road. |
| RoadInfoService(see page 40) | Determines traffic speed limits through the road identifier LinkID. |
| Open Street Map API(see page 47) | Determines traffic speed limits through the road identifier LinkID. |




Interaction of the application microservices is as follows:

1. When launching the mobile application, user authentication is performed (can be performed) by means of the Scoring Proxy microservice based on nginx.
2. Microservice DeviceInfo(see page 33) collects and stores information:

- about mobile devices of users (smartphones) - DeviceInfo (SQL) database;
- about events related to users devices - Google StackDriver database.

3. A mobile device establishes a connection with the server over the TCP. Data on mobile device activity level, telemetry data and a number of other data from a user mobile device is sent to the server for the Mobile receiver(see page 35) microservice by means of Binary data transfer protocol(see page 29).

The Mobile Receiver(see page 35) microservice distributes data received from a mobile device as follows:

4a. Data identifying the trip start is sent to the Trip triggers(see page 38) database.

4b. Telemetry data sent by a mobile device during the trip is transferred to the MobileEvents(see page 34) database.

5. It forms a set of data which is needed for identifying trip start and finish for a specified user and trasfers it to Trips Queue(see page 39) to form a queue of trips and further processing.

6. The Trip job(see page 38) microservice checks for data on trips from Trips Queue(see page 39) every second.

7. If there is available data on trips in Trips Queue(see page 39), the Trip job(see page 38) microservicce extracts this data and sends it to the Scoring API(see page 36) service for processing.

8. Once the Scoring API(see page 36) service has received data from Trip Job(see page 38) to create a trip, it creates a trip in the form of a  Trip - table(see page 36) and sends data on the trip to the Scoring (SQL)(see page 36) database for further storing.

9. After a trip has been created and saved, the Scoring API(see page 36) service sends an aknowledgement on the trip creation to the Trip Job(see page 38) microservice.

Once the Trip Job(see page 38) microservice has received an aknowledgement on the trip creation from the Scoring API(see page 36) service, it performs the following actions:

10. It transfers data on the created trip to the Scoring Queue(see page 38) microservice, which is a queue for further processing.

11. It deletes data on the created trip from Trips Queue(see page 39).

The Scoring Job (see page 38)service performs the following actions:

12. Extracts data on trips from Scoring Queue(see page 38).

13. It transfers data on trips to the Mobile Calculator(see page 33) service for further enriching with additional information and calculating a trip score.

14. Once the Mobile Calculator(see page 33) service has received data on the formed trip, it requests “raw data“ on the trip from the MobileEvents(see page 34) database.

15. The MobileEvents(see page 34) database transfers telemetry data collected during the trip period and received earlier from the Mobile receiver(see page 35) microservice to the Mobile Calculator(see page 33) service upon its request.

16. Raw telemetry data collected during the trip period is sent to the Link Mapper(see page 39) microservice. The task of the Link Mapper(see page 39) microservice is to map road identifiers (LinkID(see page 20)), over which a vehicle (smartphone) of the mobile application user was moving, to the raw trip data.

17. The Link Mapper(see page 39) microservice interacts with the OSRM(see page 40) microservice by transferring it GPS coordinates of a vehicle movement route.

18. The OSRM(see page 40) microservice analyses GPS coordinates of a vehicle route and considering allowable margin of errors in determining coordinates in the GPS system, it returns GPS coordinates of roads closest to the vehicle movement route to the Link Mapper(see page 39) mircoservice.

19. The Link Mapper(see page 39) microservice transfers GPS coordinates of the road to the Nominatim(see page 40) microservice.

20. The Nominatim(see page 40) microservice performs reverse geocoding: it determines an address of an object by its coordinates, forms a value of the LinkID(see page 20) road identifier and returns it to the Link Mapper(see page 39) microservice.

21. The Link Mapper(see page 39) microservice transfers the value of the road identifier to the Mobile Calculator(see page 33) service.

22. The Mobile Calculator(see page 33) service transfers the LinkID(see page 20) value to the RoadInfoService(see page 40) microservice.

23. The RoadInfoService(see page 40) microservice transfers the LinkID(see page 20) value to the external Open Street Map(see page 47) service.

24. The Open Street Map(see page 47) service determines a number of road characteristics for the recieved LinkID(see page 20); these particularly include speed limiting and locality tag. Therefore, the raw trip data is being enriched with data that more fully characterises separate sections of the vehicle movement. Characteristics corresponding to LinkID(see page 20) are returned to the RoadInfoService(see page 40) microservice.

25. The RoadInfoService(see page 40) microservice transfers road characteristics corresponding to the LinkID(see page 20) to the Mobile Calculator(see page 33) service for the trip score calculation.

26. The Mobile Calculator(see page 33) service processes enriched trip data obtained from the RoadInfoService(see page 40) microservice and sends resulting calculated trip data to the Scoring Job(see page 38) microservice, which includes a trip score, trip duration and length, penalties for the trip period.

27. Statistical trip data is sent from the Scoring Job(see page 38) microservice to the Scoring API(see page 36) service.

28. Statistical trip data is sent from the Scoring API(see page 36) service to the Scoring (SQL)(see page 36) database for storing.

29. The Scoring API(see page 36) service sends an aknowledgement on the trip recording into the Scoring (SQL)(see page 36) database to the Scoring Job(see page 38) microservice.

30. The Scoring Job(see page 38) microservice initiates trips removal from the queue of unprocessed trips that is supported by the Scoring Queue(see page 38) microservice.

## 2.2 Basic variables types and descriptions

Attributes of objects and their description

| Name | Type | Description |
|---|---|---|
| CompanyID | int64 | Company identifier, current value “0”. |
| DeviceID | Guid | Mobile device identifier generated randomly by the mobile application. |
| FinishTimestamp | int64 | Trip end timestamp, Unixtime. |
| Link | | Polyline which denotes a part of the road with the same properties throughout this section on the map. |

| Name | Type | Description |
|---|---|---|
| LinkID | | Identifier of a road section. |
| Safety Object | | An object near which a driver must follow “special“ behaviour rules, for example, speed limiting. |
| StartTimestamp | int64 | Trip start timestamp, Unixtime. |
| Tag | string | Field for describing a role of the user during a trip, can consist of several parts divided by commas.<br>Allowable values:<br>• a user in the role of the driver of his car - “driver_in_ my_car”<br>• a user in the role of the driver of somebody’s car - “driver_not_in_my_car”<br>• a user in the role of the passenger of a car - “passenger”<br>• a user is using public transport - “public_transport” |
| Trip | | Trip as a logic entity. |
| UserID | Guid | User identifier generated randomly by the mobile application. |
| VehicleID | string | Vehicle identifier generated randomly by the mobile application. |

# 3 Server solutions

The server part of the Open source solution Architecture(see page 17) includes:

- Network infrastructure(see page 22) of the Open source solution built on the basis of Google Cloud;
- Internal services(see page 29);
- External services(see page 41);
- Databases.

## 3.1 Network infrastructure

### 3.1.1 Infrastructure deployment


- Kubernetes cluster(see page 22)
- GIS Host(see page 23)
  - Services:(see page 23)
    - OSRM-backend(see page 23)
    - Nominatium(see page 24)
    - Test queries(see page 25)
  - Monitoring(see page 25)
  - Maps(see page 26)
  - Services working steps(see page 26)


#### 3.1.1.1 Kubernetes cluster

- **Connect string:**

```
gcloud container clusters get-credentials opensource-production --zone europe-west3-c --project
opensource-307109
```

| | |
|---|---|
| Network | opensource-production-vpc[4] |
| Subnet | opensource-production-subnet1[5] |
| Version | 1.18.12-gke.1210 |

- **Infrastructure deployment:**

git clone prod branch https://bitbucket.org/r-telematica/k8s/src/opensource/prod/

modify variables and container versions if needed

run kubectl diff -f .

4 https://console.cloud.google.com/networking/networks/details/opensource-production-vpc?project=opensource-307109
5 https://console.cloud.google.com/networking/subnetworks/details/europe-west3/opensource-production-subnet1?project=opensource-307109

Make sure everything is correct

kubectl apply -f .

#### 3.1.1.2 GIS Host

**Machine type**: e2-highmem-2 (2 vCPUs, 16 GB memory)

**Network interfaces:** opensource-production-vpc[6] gis-internal (10.10.0.11) gis-external (34.89.230.4)

**Connect string:**

```
gcloud beta compute ssh --zone "europe-west3-c" "gis" --project "opensource-307109"
```

DEV Gis[7] - swiss / ukrain / central russsian federation (*external IP: 34.89.136.76*)
PROD: gis-v2[8] - swiss / ukrain / central russsian federation (*external IP: 34.141.93.59*)
~~PROD Gis - swiss (*external IP: 34.89.230.4*)~~

Services:

**OSRM-backend**

5.22.0 /home/chient/osrm-backend sudo service osrm-routed status
http://project-osrm.org/docs/v5.24.0/api/#route-service

Disk and Memory Requirements[9]

**Pre-Processing**

For the car profile you will need around 175 GB of RAM for pre-processing and around 280 GB of STXXL disk space. You'll also need 35 GB for the planet `.osm.pbf` file, and 40-50 GB for the generated datafiles.

**Runtime**

For the car profile you will need around 64GB of RAM.

We basically just load all the files into memory, so whatever the output file size from pre-processing - that's roughly how much RAM you'll need (minus the size of the `.fileIndex` file, which is `mmap()`-ed and read on-demand).

Install Instructions: https://github.com/Project-OSRM/osrm-backend/wiki/Building-OSRM
Downloaded map data: https://download.geofabrik.de/europe[10]
https://download.geofabrik.de/europe/switzerland-latest.osm.pbf

OSRM installation instruction (with map)

*Run as service:*

6 https://console.cloud.google.com/networking/networks/details/opensource-production-vpc?project=opensource-307109
7 https://console.cloud.google.com/compute/instancesDetail/zones/europe-west3-c/instances/gis?authuser=1&organizationId=676511452833&project=development-282908&rif_reserved
8 https://console.cloud.google.com/compute/instancesDetail/zones/europe-west3-c/instances/gis-v2?authuser=1&project=opensource-307109&rif_reserved
9 https://github.com/Project-OSRM/osrm-backend/wiki/Disk-and-Memory-Requirements
10 https://download.geofabrik.de/europe/switzerland-latest.osm.pbf

```
nano /etc/systemd/system/osrm-routed.service

[Unit]
Description=Open Source Routing Machine
Wants=network-online.target
After=network.target network-online.target

[Service]
ExecStart=/usr/local/bin/osrm-routed --threads=4 /home/chient/osrm-backend/merged-maps.osrm
User=chient
Group=chient
Restart=always
RestartSec=5s

[Install]
WantedBy=multi-user.target

sudo systemctl start osrm-routed
sudo systemctl status osrm-routed
sudo systemctl enable osrm-routed

# Show logging
sudo journalctl -eu osrm-routed
# Show logging in realtime
sudo journalctl -u osrm-routed -f
```

Nominatium

3.6.0 /srv/nominatium/ sudo service apache2 status https://nominatim.org/release-docs/latest/api/Reverse/#parameters

Install Instructions https://blog.coffeebeans.at/archives/1154

Config file: /etc/apache2/conf-available/nominatim.conf

```
<Directory "/srv/nominatim/build/website">
  Options FollowSymLinks MultiViews
  AddType text/html   .php
  DirectoryIndex search.php
  Require all granted
</Directory>
```

How to check service:

```
curl -X GET 'http://localhost/nominatim/reverse?format=jsonv2&lat=47.1718168&lon=8.5318365'
```

Downloaded map data: `https://download.geofabrik.de/europe/switzerland-latest.osm.pbf`

Nominatem installation instruction (with map)

Test queries

```
-- CH
-- matching
http://[IP]:5000/match/v1/driving/
8.5132685,47.1929169;8.5122919,47.1911888;8.5097866,47.1920052;8.5070324,47.1918297;8.5050898,47.1905670;8.
5031261,47.1891365;8.5012503,47.1877251;8.4986401,47.1871758;8.4972801,47.1863708;8.4966402,47.1859703;8.49
39280,47.1862259?
radiuses=14.2179499;21.1710072;10.5064726;11.5851946;10.9675856;10.9916372;7.1024551;24.4760647;16.8963928;
15.4699860;5.0728993&timestamps=1619523085;1619523120;1619523134;1619523149;1619523164;1619523178;161952319
6;1619523223;1619523231;1619523234;1619523267&geometries=geojson&tidy=true&annotations=true&steps=true

-- route
http://[IP]:5000/route/v1/driving/9.11675,47.124403;9.187389,47.114006?
overview=false&geometries=geojson&steps=true

-- nominatem
http://[IP]/nominatim/reverse?format=jsonv2&lat=47.170004&lon=8.521052&zoom=17

-- MSK
-- route
http://[IP]:5000/route/v1/driving/37.649296,55.828679;37.637541,55.811645?
overview=false&geometries=geojson&steps=true

-- nominatem
http://[IP]/nominatim/reverse?format=jsonv2&lat=55.888500&lon=37.738721&zoom=17
```

Monitoring

Google Cloud Monitoring (CPU & Memory) Link[11]

Alerts: GIS-PROD-OPENSOURCE-CPU[12] GIS-PROD-OPENSOURCE-MEM[13]

Custom Alert and restart on error:

```
sudo -i
cd /home/chient/osrm-backend
nano alert.sh

#!/bin/bash
curl -f -o /dev/null "http://localhost:5000/route/v1/driving/9.11675,47.124403;9.187389,47.114006?
overview=false&geometries=geojson&steps=true" || curl -X POST --data-urlencode "payload={\"channel\":
\"#alerts\", \"username\": \"webhookbot\", \"text\": \"RESTART osrm-routed service on $HOSTNAME\",
\"icon_emoji\": \":ghost:\"}" https://hooks.slack.com/services/TAAASBXKL/B01RQRE972B/
pRJnw9YljUeXir140fpmaDSJ  && systemctl start osrm-routed
curl -f -o /dev/null "http://localhost/nominatim/reverse?format=jsonv2&lat=47.170004&lon=8.521052" || curl
-X POST --data-urlencode "payload={\"channel\": \"#alerts\", \"username\": \"webhookbot\", \"text\":
```

11 https://console.cloud.google.com/compute/instancesMonitoringDetail/zones/europe-west3-c/instances/gis?project=opensource-307109&tab=monitoring&duration=PT1H
12 https://console.cloud.google.com/monitoring/alerting/policies/12306319785175557O5?project=opensource-307109
13 https://console.cloud.google.com/monitoring/alerting/policies/7501611835422453 4?project=opensource-307109

```
\"RESTART nominatim service on $HOSTNAME\", \"icon_emoji\": \":ghost:\"}" https://hooks.slack.com/services/
TAAASBXKL/B01RQRE972B/pRJnw9YljUeXir140fpmaDSJ  && systemctl restart apache2.service


# optional but available step: chmod ugo+x /home/chient/osrm-backend/alert.sh

crontab -l | { cat; echo "*/1 * * * * bash /home/chient/osrm-backend/alert.sh >/dev/null 2>&1"; } | crontab
-
crontab -l

For test run
systemctl stop osrm-routed && systemctl stop apache2.service
white 1 min and see alert and running services
```

**Maps**

https://wiki.openstreetmap.org/wiki/Osmconvert

$ sudo apt install osmctools
$ osmconvert albania-latest.osm.pbf --out-o5m | osmconvert - cyprus-latest.osm.pbf -o=merge.pbf

$ osmconvert albania-latest.osm.pbf --out-o5m | osmconvert - cyprus-latest.osm.pbf | osmconvert - ukraine-latest.osm.pbf -o=merge.pbf

**Notes:** *There are no full-fledged instructions and automatic scripts for deploying services, it is necessary to containerize, the services have a high load capacity*

Copy from remote VM: gcloud compute copy-files gis-osrm:/srv/nominatim/mapdata ~/ --zone "europe-west3-c" --project "normalsigma"

Services working steps

1. LinkMapper: calls the osrm api on the gis host (for matching & routing)
   a. Before matching we make filtering: points with accuracy > 100 should be removed
   b. We do map matching by batches (due to URI length limitation in HTTP request)
      i. then we take waypoint for LinkToPoints object, geometry of Matching for GeoLinkInfo.
         waypoint - the mapped point based on the base point
         geometry - simple route (array of points) of the matching result
   c. After matching we call routing service:
      i. we build routing between batch points result of map matching, when we receive the result from routing we verify do we need to take points from routing(RULE: we will take the distance of routing if it's equal or greater not more then 5% then matching distance.) or leave the points from matching geometry.
2. LinkMapper: calls the nominatem api on the gis host (example - https://nominatim.openstreetmap.org/search?q=26.9617265,%20-81.983976&format=json&addressdetails=1
   for osm_id
3. RoadInfo: calls the OSM for getting speed limits the external API https://www.openstreetmap.org/api/0.6/way/805832330/full.json

## 3.1.2 OSRM Map update

### 3.1.2.1 *Preparation*

Please change user before any operatinons:

```
su chient
```

Please, disable cron task alert before OSRM and Nominatem stack updating:

```
cronrab -e
```

and comment as:

```
#*/2 * * * * bash /home/chient/osrm-backend/alert.sh >/dev/null 2>&1
```

### 3.1.2.2 OSRM

If not installed a tool for map merge

```
apt install osmctools
```

Get maps

```
wget -O switzerland-exact.osm.pbf https://planet.osm.ch/switzerland-exact.osm.pbf
wget -O ukraine-exact.osm.pbf https://download.geofabrik.de/europe/ukraine-latest.osm.pbf
wget -O cfd-exact.osm.pbf https://download.geofabrik.de/russia/central-fed-district-latest.osm.pbf
wget -O germany-exact.osm.pbf https://download.geofabrik.de/europe/germany-latest.osm.pbf
```

Map merge

```
osmconvert ukraine-exact.osm.pbf --out-o5m | osmconvert - switzerland-exact2.osm.pbf | osmconvert - cfd-
exact.osm.pbf -o=merged-maps.pbf
```

Extract the routing data

```
osrm-extract merged-maps.pbf -p profiles/car.lua
```

Creating a Hierarchy

```
osrm-contract merged-maps.osrm
```

Launching

```
osrm-routed merged-maps.osrm
```

*Fix permissions*

```
chown -R chient:chient /home/chient/osrm-backend/
chmod -R a+x /home/chient/osrm-backend/
```

#### 3.1.2.3 Nominatim

```
sudo -i
sudo -u $USERNAME dropdb nominatim
```

*Dropping db is the temporary solution:* https://nominatim.org/release-docs/3.2.0/admin/Import-and-Update/#updates

Variables

```
export USERNAME=nominatim
export USERHOME=/srv/nominatim
export NOMATIM_VERSION=3.6.0
export NOMINATIM_SOURCE_DIR=$USERHOME/Nominatim-${NOMATIM_VERSION}
export BUILD_LOCAL=$USERHOME/build/settings/local.php
export BUILD_UTILS=$USERHOME/build/utils
export PG_CONF=/etc/postgresql/10/main
export MAP_DATA_DIR=$USERHOME/mapdata
```

Copy merged map (from osrm dir to nominatim) to `MAP_DATA_DIR`

```
cp merged-maps.pbf $MAP_DATA_DIR
```

Go to `MAP_DATA_DIR` and fix permissions after getting all data

```
cd $MAP_DATA_DIR
chmod -R a+x $USERHOME
chown -R ${USERNAME}:${USERNAME} $USERHOME
```

Install map data

```
sudo -u $USERNAME -H sh -c "$BUILD_UTILS/setup.php --osm-file $MAP_DATA_DIR/merged-maps.pbf --all" 2>&1 |
tee $USERHOME/setup-merged.log
```

*Enable cron task alert before OSRM and Nominatem stack updating*

```
cronrab -e

#*/2 * * * * bash /home/chient/osrm-backend/alert.sh >/dev/null 2>&1
```

## 3.2 Internal services

### 3.2.1 Client-server data exchange

Data exchange between the mobile application and the server is performed using the **Binary data transfer protocol v.12.1**:

- Data is transferred via TCP protocol.
- Data is transferred in the binary form; compression and coding are not applied.
- Byte order - Little-Endian.

---

Conceptually, data transfer can be divided into the following steps:

1. Mobile device establishes a TCP connection with the server.
2. The server launches a waiting cycle for the **events** from the device.
3. The first message after establishing a connection - authorization of data transmission (type 0).
4. Getting a response from the server - type 5 messages.
5. The second message - metadata transmission (type 1) - later on, upon metadata change, type 1 message can be sent any time.
6. Getting a response from the server - type 5 message.
7. Type 1, 2, 3, 4 or 6 message transmission - waiting for the server to respond after each message - type 5 messages.
8. Terminating a TCP-connection by a client or a server.

---

A mobile device transmits every separate **event** in the following way:

1. Fixed-size **token** transmission.

2. Message body transmission; the server determines a message body size from a **token.**

---

After every event transmission, the client waits for ACK (type 5) message from the server; the next message is **not sent** by the client **until a response (successful) is received** from the server. If the server does not respond for a defined period of time, then the client **does not repeat** the previous message, but **terminates the TCP session** and re-establishes it. If the server responds with the body 0x01 (fail) ACK message, then error determining depends on the context.

---

**The token** has a fixed size, currently - **4 bytes**

| Byte number | Description |
|---|---|
| 0 | Message timestamp; last version - **12 (0x0C)** |
| 1 | Message type (on its basis the server determines the message body size) |
| 2 | Mobile OS type (0x01 - iOS; 0x02 - Android) |
| 3 | Not used |

---

The following **message types** are supported:

| Type | Description | Message body length | Data type | Contents |
|---|---|---|---|---|
| 0 | Authorization | not fixed | byte[] | See bellow |
| 1 | Metadata transfer | not fixed | byte[] | See bellow |
| 2 | Trip start | 4 bytes | uint_32 | UNIX Timestamp* |
| 3 | Telematic event | 36 bytes | custom | See bellow |
| 4 | Trip finish | 4 bytes | uint_32 | UNIX Timestamp* |
| 5 | ACK | 1 byte | byte | 0x00 - in case of success, 0x01 - in case of fail |
| 6 | Message package | not fixed | custom | Package consisting of type 1, 2, 3 or 4, 7, 8 messages |
| 7 | Accelerometer | 20 bytes | byte[] | for Open Source |

| Type | Description | Message body length | Data type | Contents |
|---|---|---|---|---|
| 8 | AccelerometerPenalty | 28 bytes | byte[] | for Open Source |

**Authorization** event (type 0), the fields follow one after another:

| | Description | Byte qty | Data type |
|---|---|---|---|
| 1 | Message length (bytes) | 2 | uint_16 |
| 2 | Message | (from field 1) | byte[] (UTF8 string) |

**Metadata** (type 1), the fields follow one after another:

<table>
<tr><th></th><th>Description</th><th>Byte qty</th><th>Data type</th></tr>
<tr><td>1</td><td>Message length (bytes)</td><td>2</td><td>uint_16</td></tr>
<tr><td>2</td><td>Message<br>
<pre><code>// example
{
"deviceId": "A630BEB2-711C-458D-8684-EC04171E4992",   // GUID value
"userId": "0BC6766B-2274-4C3D-87E6-281916A580D3",     // GUID value
 "vehicleId": "6F9619FF-8B86-D011-B42D-00CF4FC964FF", // GUID value
 "tag": "driver_in_my_car",                           // string tag name
 "isBluetoothOn": "0"                                // 0 or 1 value
}</code></pre></td><td>(from field 1)</td><td>byte[]<br>(String in UTF8)</td></tr>
</table>

**Telematics event** contents (type 3), the fields follow one after another:

| Data type | Description |
|---|---|
| uint_32 | UNIX Timestamp* of the event |
| sint_32 | Latitude multiplied by $10^7$ |

| Data type | Description |
|---|---|
| sint_32 | Longitude multiplied by $10^7$ |
| float_32 | Speed in meters per second |
| float_32 | X-axis acceleration |
| float_32 | Y-axis acceleration |
| float_32 | Z-axis acceleration |
| float_32 | Azimuth (direction of movement) |
| float_32 | HDOP (horizontal dilution of precision) |
| uint_32 | startCandidateByBluetooth  for each trip coordinate received:<br>• if Bluetooth device detected - startCandidateByBluetooth = 1,<br>• if Bluetooth device not detected - startCandidateByBluetoothif = 0. |
| uint_32 | startValidate for each trip coordinate received:<br>• if trip start is validated by activity - startValidate = 1,<br>• if trip start is not validated by activity - startValidate = 0. |

*UNIX Timestamp - seconds form ***1970-01-01T00:00:00Z*** time

Message package (type 6):

| | Description | Byte qty | Data type |
|---|---|---|---|
| 1 | Qty of messages in the array | 2 | uint_16 |
| 2 | Messages array ( in each token+body) | (depends on the type) | custom |

### 3.2.2 Scoring Proxy - authentication

Authentication of a mobile application user in the Open source solution is performed through the “Basic authentication“ method. For “Basic authentication“ to be performed, a user receives a pair of values `<username>:<password>` after contacting the Technical support(see page 7).

Once the mobile application of the Open source solution has been installed on the user's smartphone, the mobile application randomly generates three identifiers:

- UserID(see page 20) - unique user identifier;

- DeviceID(see page 20) - mobile device identifier;
- VehicleID(see page 20) - vehicle identifier.

The identifiers UserID(see page 20), DeviceID(see page 20) and VehicleID(see page 20) are used both in the interaction of the mobile application with the server and in the interaction of the internal microservices of the Open source solution with each other.

### 3.2.3 DeviceInfo

The microservice is designed for processing and storing information:

- about users' mobile devices - DeviceInfo (SQL)(see page 33) database;
- about events related to users' devices - Google StackDriver(see page 41) database.

### 3.2.4 DeviceInfo (SQL)

The DeviceInfo (SQL) database stores information on users' mobile devices.

| Name | Type | Description |
|---|---|---|
| platform | string<br>Enum:<br>['Undefined', 'Ios', 'Android'] | Mobile device platform |
| model | string | Mobile device model |
| manufacturer | string | Mobile device manufacturer |
| osVersion | string | Version of mobile device OS |
| deviceId | string | Mobile device identifier |
| appVersion | string | Version of installed application |
| pushId | string | Firebase token identifier for sending push notifications |
| AppsFlyerId | string | AppsFlyer identifier |

### 3.2.5 Mobile Calculator

The service performs the following tasks:

- Calculating a trip score on demand of Scoring Job(see page 38).
- Requesting DeviceGUID(see page 20), StartTimestamp, FinishTimestamp values from Scoring Job(see page 38).

- Using the received information, Mobile Calculator requests raw trip data from the MobileEvents (no SQL)(see page 34) database.
- All received data on trip events is sent to LinkMapper(see page 39) microservice.
  - LinkMapper(see page 39) is a service which for every point returns a Link - a polyline that on the road map denotes a road section with the same properties.
  - for received Link(see page 20), requests are sent to the RoadInfo(see page 40) service; the following data is sent in response for these requests for each Link:
    - Road:
      - speedLimit - speed limit on a road section
      - streetName - name of a street
      - safetyObjects - an array of SafetyObjects. SafetyObject is an object within which a driver should obey certain behaviour rules, for exapmle, speed limit
    - IsInCity - locality tag
- Through coopereation with LinkMapper(see page 39) and RoadInfo(see page 40) microservices the Mobile Calculator supplies a trip (each point) with the following data:
  - linking to road properties through the LinkMapper(see page 39) service
  - data from the RoadInfo(see page 40) service:
    - speed limits on a road section;
    - locality tag.
- Once all events have been applied to current road conditions, the events are grouped into sequences and subsequences of a trip.
- Events of sharp acceleration and sharp deceleration are being idetified. An event is identified as a sharp acceleration or a sharp decelartion by means of logical multiplication operation for a series of parameters (condition **AND**):
  - Exaclty 1 second should pass between events
  - **AND** both events have a field Accuracy <= 10
  - **AND** (Heading > 0 || OR SpeedKph >0)
  - **AND** Heading of two events differs in less than 30°
- Acceleration threshold values, the difference in speed of a following event and an event being considered with the interval of 1 second between the events (kilometre / hour per second):
  - >= 14.11 - Acceleration event of III degree
  - >= 12.35 - Acceleration event of II degree
  - >= 10.58 - Acceleration event of I degree
- Deceleration threshold values, the difference in speed of an event being considered and the following one with the interval of 1 second between the events (kilometre / hour per second):
  - >= 19.44 - Acceleration event of IV degree
  - >= 15.91 - Acceleration event of III degree
  - >= 12.35 - Acceleration event of II degree
  - >= 11.66 - Acceleration event of I degree
- If there are gaps in the data, the Mobile Calculator evaluates data fragments separately and then it produces an average distance weighted values as a score of the whole trip.
  The resulting scroing statistics of a trip contains the following data:
  - Weighted average Accidentness
  - Weighted average score ScorePrecent
  - Total duration
  - Total distance
  - List of violations
- The Mobile Calculator requests the Scoring API(see page 36) service to add the calculated score to its trip.

### 3.2.6 Mobile Events Database

NoSQL MobileEvents Table stores all data packages received from a mobile device.

| Field | Data type | Description |
|---|---|---|
| PartitionKey | | Primary key. The value is constructed according to the pattern [YYYY]-[MM]_[DeviceGuid]<br>Year and month shoud be indicated for partitioning. |
| RawKey | bigint | Secondary key. The value is generated according to the formula 9999999999 - UnixTime. It is needed for sorting in the descending order. |
| Timestamp | bigint | Event timestamp, entered while recording into the database. |
| Type | string | [ TELEMETRY \| START \| STOP ] |
| UnixTime | bigint | Event time, sent from a mobile phone.<br>If Type = "TELEMETRY", then the GPS system time is saved;<br>If Type = "START \| STOP", then the smartphone system time is saved. |
| Latitude | float32 | Latitude. |
| Longitude | float32 | Longtitude. |
| SpeedMps | float32 | Speed, Mps. |
| Heading | float32 | Heading direction, degrees.<br>0 degrees = North, 90 = East, 180 = South, 270 = West. |
| Accuracy | float32 | Accuracy, received as a part of GPS system data . |

Sampling is done by PartitionKey and RawKey.

### 3.2.7 Mobile receiver

The Mobile Receiver service is designed to receive telemetry data from the mobile application and performs the following tasks:

- Controlling a connection between the mobile application and the server over the TCP protocol.
- Receiving data from the mobile application over the Binary data transfer protocol.
- Recording data on all events into the Mobile Events(see page 34) Table.
- Recording data on the ‘Start’ event, corresponding to the beginning of movement, to the Trip Triggers(see page 38) table.
- If the 'Stop(see page 121)' event occurs, it determines a moment when the 'Start(see page 121)' event has not been recorded for a mobile terminal through its DeviceID(see page 20), forms a set of telemetry data which have been received within the time period between 'Start(see page 121)' and 'Stop(see page 121)' events.

- Transferring a prepared telemetry data set as well as values of DeviceID(see page 20), UserID(see page 20), VehicleID(see page 20), Tag(see page 20), StartTimestamp(see page 20), FinishTimestamp(see page 20) in the json format to Trips Queue(see page 39).
- Accounting all performed operations (logging).

### 3.2.8 Scoring API - Mobile

The service performs the following tasks:

- Receiving UserID(see page 20), DeviceID(see page 20), VehicleID(see page 20) from the Scoring Proxy microservice and transferring them to the DeviceInfo(see page 33) microservice for processing and storing.
- Receiving data from Trip Job(see page 38) for creating a trip, creating a trip in the form of a table Trip - table(see page 36) that is stored in the Scoring (SQL)(see page 36) database.
- Confirming trip creation for Trip Job(see page 38) thus triggering the mechanism for further raw trip data processing.
- Receiving the final calculated trip data from ScoringJob(see page 38), transferring it to the Scoring (SQL)(see page 36) database for storing .
- After saving calculated trip data via ScoringJob(see page 38) microservice, it initiates Scoring Queue(see page 38) clearing.

### 3.2.9 Scoring (SQL)

The SQL database that stores trip records in the following tables:

| Table | Description |
|---|---|
| Penalty | List of records on violations during a trip |
| ScoringInfo | Trip score |
| Trip | List of trips |

**Structure of Tables .**

Penalty - table

| **Field** | **Data type** | **Description** |
|---|---|---|
| **TripId** | binary(16) | Trip identifier, foreign key |
| Timestamp | bigint | Time of violation start in Unix format |
| Type | nvarchar(50) | Violation type, for example, speeding, acceleration. |
| DurationMs | bigint | Duration of violation in milliseconds |

| Field | Data type | Description |
|---|---|---|
| Value | float | The number of penalty points calculated in Mobile Calculator(see page 33) |

ScoringInfo - table

| Field | Data type | Description |
|---|---|---|
| **TripId** | binary(16) | Trip identifier, foreign key |
| ScorePercent | float | Score (%) |
| DurationSec | bigint | Trip duration based on scoring data (events), in seconds.<br>Sum of trip parts durations. |
| DistanceMeters | bigint | Trip distance based on scoring data (events), in meters.<br>Sum of trip parts distances. |
| Accidentness | float | Average trip accidentness. |

Trip - table

| Field | Data type | Description |
|---|---|---|
| **TripId** | binary(16) | Trip identifier, created when creating a trip. |
| UserId | binary(16) | User identifier. |
| DeviceId | binary(16) | Mobile device identifier. |
| VehicleId | binary(16) | Vehicle identifier. |
| StartTimestamp | bigint | Trip start time, Unixtime |
| FinishTimestamp | bigint | Trip finish time, Unixtime |
| Tag | nvarchar(50) | Trip tag |

### 3.2.10 Scoring Job

The service performs the following tasks :

- Extracting raw data on trips from Scoring Queue(see page 38).
- Sending data on a trip to the Mobile Calculator(see page 33) service for enriching and calculating the score.
- Invoking the Scoring API(see page 36) service and transferring final calculated trip data to it.
- Initiating Scoring Queue(see page 38) clearing on Scoring API(see page 36) request.

### 3.2.11 Scoring Queue

The service is represented by a queue of trips for evaluation and enriching with data. Data on trips is received from Trip Job(see page 38) and sent to Scoring Job(see page 38).

The Message field contains json with the following structure.

| Name | Type | Description |
|---|---|---|
| TripId | Guid | Trip identifier. |
| DeviceId | Guid | Device identifier. |
| From | long | Start (UNIX timestamp). |
| To | long | Finish (UNIX timestamp). |

### 3.2.12 Trip job

The service generates sets of data about the trip:

- extracts initial trip data from Trips Queue(see page 39) with the frequency of 1Hz (1 time per second);
- if Trips Queue(see page 39) has data on a trip, then Trip Job extracts this data for processing;
- invokes Scoring API(see page 36) service, transfers the data (telemetry, DeviceID(see page 20), UserID(see page 20), VehicleID(see page 20), Tag(see page 20), StartTimestamp(see page 20), FinishTimestamp(see page 20)) to it for trip generating;
- once Scoring API(see page 36) has acknowledged the trip generation, the Trip job service sends data to Scoring Queue(see page 38);
- clears trip data from Trips Queue(see page 39).

### 3.2.13 Trip triggers database

The service is a noSQL database for storing data on events identifying the trip start.

Stored data

| Field | Data type | Description |
|---|---|---|
| PartitionKey | | Primary key. The value is constructed according to the pattern [YYYY]-[MM]_[DeviceGuid]<br>Year and month should be indicated for partitioning |
| RawKey | bigint | Secondary key. The value is always "START" |
| Timestamp | bigint | Event timestamp, entered while recording to the database |
| UserId | binary(16) | User identifier in the company. |
| DeviceId | binary(16) | Device identifier. |
| VehicleId | binary(16) | Vehicle identifier. |
| StartTimestamp | bigint | Trip start, Unixtime |
| Tag | nvarchar(50) | Trip tag. |

### 3.2.14 Trips Queue

The service is designed to generate queues of initial data on a trip:

- From Mobile Receiver(see page 35) it receives data for trip creation in the json format: DeviceID(see page 20), UserID(see page 20), VehicleID(see page 20), Tag(see page 20), StartTimestamp(see page 20), FinishTimestamp(see page 20).
- It generates and process a queue using NATS.
- It cooperates with the Trip Job(see page 38) service, which forms a data set for trip creation based on data in the queue.

### 3.2.15 LinkMapper

A microservice with a task of snapping GPS coordinates of vehicle movement points to an identifier of the nearest road or a road section (linkID(see page 20)), along which a vehicle is moving.

The LinkMapper microservice receives GPS coordinates of vehicle movement points from the Mobile Calculator(see page 33) service and transfers them to the OSRM(see page 40) microservice, which returns GPS coordinates of the roadway nearest to the car route to the LinkMapper microservice.

The LinkMapper microservice transfers GPS coordinates of the road received from the OSRM(see page 40) microservice to the Nominatim(see page 40) microservice, which snaps received GPS coordinates of the road to linkID(see page 20) and returns its value to the LinkMapper microservice.

### 3.2.16 Nominatim

Nominatim[14] - open-source geocoding service. Nominatim uses OpenStreetMap (OSM) data to find locations on Earth by name and address (geocoding). It can also do the reverse, find an address for any location on the planet.

Nominatim API has the following endpoints for querying the data:

- /search - search OSM objects by name or type;
- /reverse - search OSM object by their location;
- /lookup - look up address details for OSM objects by their ID;
- /status - query the status of the server;
- /deletable - list objects that have been deleted in OSM but are held back in Nominatim in case the deletion was accidental;
- /polygons - list of broken polygons detected by Nominatim;
- / details - show internal details for an object (for debugging only).

This project uses the reverse geocoding[15] and polygons functions of the Nominatim service, i.e. it searches for an OSM object by the object's location. Reverse geocoding generates a road address from a latitude and longitude and then forms an identifier of the road section - linkID(see page 20).

Integration of the Nominatim service into this project is described in the section Infrastructure deployment(see page 22).

### 3.2.17 RoadInfo Service

It is a microservice that provides system's interaction with an external service that allows receiving additional information on the road, along which a vehicle is moving: locality tag, speed limits, etc.

In the current system implementation RoadInfo Service interacts with the external Open Street Map API(see page 47) service. RoadInfo Service receives linkID(see page 20) from the Mobile Calculator(see page 33) service and sends an query to Open Street Map API(see page 47) for snapping the additional information about the road (locality tag, speed limits, etc.) to the indicated linkID(see page 20). Data received from Open Street Map API(see page 47) is returned to the Mobile Calculator(see page 33) service.

### 3.2.18 OSRM

Open Source Routing Machine (OSRM[16]) - routing engine for shortest paths in road networks.

The following services are available via HTTP API, C++ library interface and NodeJs wrapper:

- Nearest - Snaps coordinates to the street network and returns the nearest matches;
- Route - Finds the fastest route between coordinates;
- Table - Computes the duration or distances of the fastest route between all pairs of supplied coordinates;
- Match - Snaps noisy GPS traces to the road network in the most plausible way;
- Trip - Solves the Traveling Salesman Problem using a greedy heuristic;
- Tile - Generates Mapbox Vector Tiles with internal routing metadata.

This project uses the Nearest[17] service: snaps a smartphone GPS coordinate to the street network and returns the nearest roads n matches.

---

14 https://nominatim.org
15 https://nominatim.org/release-docs/develop/api/Reverse/
16 http://project-osrm.org
17 http://project-osrm.org/docs/v5.23.0/api/#nearest-service

Integration of the OSRM service into this project is described in the section Infrastructure deployment(see page 22).

## 3.3 External services

### 3.3.1 Google StackDriver

Google StackDriver[18] - centralised cloud computing system to uptime monitoring, log analysis, error reporting and production debugging, across Google Cloud Platform and Amazon Web Services. It provides performance and diagnostics data to public cloud users.

Google StackDriver database stores information on events captured on users' mobile devices.

- Which information is transferred:
  - meta section - general information about a device;
  - checks section - event data on the availability of different sensors and device services;
  - logs section - diagnostic events of a device, including **errors**, **warnings**, **info**.
- Where the information is saved:
  - Stackdriver saves records, consisting of the **meta** section and one of the **checks, logs** elements, in the form of JSON Payload 3. One query can transfer a random number of **checks, logs** elements. Each element is saved together with the **meta** element as a separate record.
  - In Stackdriver **logs** are saved into containers called ‘RTSTart.DeviceInfo.MobileLogger’ + ‘Staging | Dev’.
- If there is a record about the device in the database, then it is updated, if not, the record is created.
- If the **userId** value is known, then the method updates information about the user's device based on data from the **meta** section.

**Data model of meta section.**

<table>
<tr><th>Name</th><th>Type</th><th>Description</th></tr>
<tr><td>version</td><td>integer($int64)</td><td>Version number of logging protocol</td></tr>
<tr><td>appVersion</td><td>integer($int64)</td><td>Version number of application</td></tr>
<tr><td>attr</td><td>
<table>
<tr><th>Name</th><th>Type</th><th>Description</th></tr>
<tr><td>platform</td><td>string
<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Undefined</td><td>Undefined</td></tr>
</table>
</td><td>Platform</td></tr>
</table>
</td><td>Attributes</td></tr>
</table>

18 https://cloud.google.com/products/operations

| Name | Type | | | | Description |
|---|---|---|---|---|---|
| | | Ios | iOS | | |
| | | Android | Android | | |
| | osVersion | integer($int64) | | OS version number | |
| | model | string | | Device model | |
| | manufacturer | string | | Device manufacturer | |
| | userId | string($uuid) | | User,<br>UserGUID | |
| | vehicleId | string($uuid) | | Vehicle<br>VehicleGUID | |
| | deveceId | string($uuid) | | Device<br>DeviceGUID | |
| | telemetryVersion | integer($int64) | | Telemetry library version | |

**Data model of check section.**

| Name | Type | Description |
|---|---|---|
| startAt | integer($int64) | Event start time, UnixTimeStamp |
| **name** | string | Event name. |
| correlationId | string($uuid) | System correlation identifier |
| **error** | boolean | Is the event an error? |
| info | string | Additional infomation |

**error**:

true - from the startAt moment the application is idle because of a **name** event, additional information is in **info**. The idle state remains until the moment startAt of an event with the same **name** that has an **error: false** or till startAt of the event STATUS_OK. In case of application stopping and restarting, the idle time interval is calculated from startAt of the last recorded event to startAt of the event  APP_CREATE.

false - the application is idle.

List of possible **name** values in the data model of the **check** section.

| Name | Description |
|---|---|
| STATUS_GPS_ENABLED | GPS status - true, false |
| STATUS_GPS_MODE | GPS mode - high accuracy, gps only, low accuracy. |
| STATUS_GPS_PERMISSION | Granted, denied |
| STATUS_STORAGE_PERMISSION | Granted, denied |
| STATUS_BT | BT Status |
| STATUS_POWER_SAFE | PowerSafe status |
| STATUS_TELEMETRY | Is telemetryService working? |
| STATUS_DETECTOR | Is detectorService working? |
| STATUS_ERROR | Error, application is in idle state |
| APP_CREATE | App Created |
| APP_STOP | App stopped |
| APP_CRASH | App crashed |
| APP_PID | pid of app process |
| STATUS_OK | All systems are functioning, no errors |
| STATUS_LOCATION | Current location |
| STATUS_BATTERY | Battery status |

| Name | Description |
|---|---|
| STATUS_RAM | RAM usage |
| DEVICE_INFO | Device info |

Data model of the **logs** section.

| Name | Type | Description |
|---|---|---|
| severity | string | Critical \| Error \| Warning \| Info \| Debug<br>levels in Stackdriver |
| message | string | Information part |
| exception | string | Saving stack trace error |
| area | string | url of the request<br>• Not used, provided for additional message filtering. |
| code | integer($int64) | http status codes[19]<br>or another code or 0 |
| **eventName** | string | Event name. |
| executionTime | integer($int64) | Execution time.<br>not used. |
| correlationId | string($uuid) | x-request-id from request<br>system correlation identifier |
| responseId | string($uuid) | x-request-id from response or empty |
| startAt | integer($int64) | Event start time, Unix timestamp |

List of possible **eventName** values in the data model of the **logs** section.

19 https://ru.wikipedia.org/wiki/%D0%A1%D0%BF%D0%B8%D1%81%D0%BE%D0%BA_%D0%BA%D0%BE%D0%B4%D0%BE%D0%B2_%D1%81%D0%BE%D1%81%D1%82%D0%BE%D1%8F%D0%BD%D0%B8%D1%8F_HTTP

| Event name | Description |
| --- | --- |
| DETECTOR_SERVICE_START | Service Started |
| DETECTOR_SERVICE_STOP | Service Stopped |
| DETECTOR_SERVICE_DESTROY | Service Destroyed |
| DETECTOR_SERVICE_LOW_MEM | Service low memory |
| DETECTOR_ACTIVITY | Check of current user activity - most probable activity |
| DETECTOR_EVENT | Detector event |
| BT_CONNECTED | BT device from detector settings connected |
| TRIP_START_CANDIDATE | Trip Start Candidate Detected |
| TRIP_STOP_CANDIDATE | Trip Stop Candidate Detected |
| TRIP_START_VALID | Trip start validation succeed |
| TRIP_START_INVALID | Start validation failed |
| TRIP_STOP_VALID | Stop validation succeed |
| TRIP_STOP_INVALID | Stop validation failed |
| TELEMETRY_RECEIVER_START_CANDIDATE | Telemetry receiving started for start candidate |
| TELEMETRY_RECEIVER_STOP_CANDIDATE | Telemetry receiving data for stop candidate |
| TELEMETRY_RECEIVER_STOP_RECEIVE | Stop receiving data |
| TELEMETRY_RECEIVER_START_RECEIVE | Start receiving data |
| TELEMETRY_RECEIVER_START_VALIDATION | Moving/deleting tmp points depending on the result of validation |
| TELEMETRY_RECEIVER_STOP_VALIDATION | Moving/deleting tmp points |

| Event name | Description |
|---|---|
| TELEMETRY_SENDER_START | Telemetry sending started |
| TELEMETRY_SENDER_STOP | Telemetry sending stopped |
| TELEMETRY_PACKAGE_SENT | Telemetry data was successfully sent to server |
| TELEMETRY_PACKAGE_ERROR | Issue sending data |
| TELEMETRY_SENDER_DONE | All data sent |
| TELEMETRY_TIME_DELTA | Difference between gps and local time - time in ms |
| TELEMETRY_ERROR | Error |
| TELEMETRY_METADATA | metadata for telemetry receiver |
| TELEMETRY_POINT | Telemetry point received |
| TELEMETRY_FIRST_POINT | First point of current trip received. Using gps time for correction |
| TELEMETRY_SERVICE_START | Service Started |
| TELEMETRY_SERVICE_STOP | Service Stopped |
| TELEMETRY_SERVICE_DESTROY | Service Destroyed |
| TELEMETRY_SERVICE_LOW_MEM | Service low memory |
| BT_DISCONNECTED | BT device disconnected |
| DETECTOR_FATAL_ERROR | Error which makes further work of detector impossible |
| DETECTOR_SETTINGS | Settings of trip detector. (On start or on settings change) - current settings |
| LOG_LEVEL | Log level (On start or on change) - current level |

| Event name | Description |
|---|---|
| OTHER | Other logs |
| SCREEN | User entered screen {number} |
| WAKE_UP_PUSH | Silent-push received |
| EXIT_REGION | User left geolocation region |
| SIGNIFICANT_LOCATION_CHANGE | Significant change in geolocation |
| TRIP_CREATE | Trip created |

### 3.3.2 Open Street Map API

OpenStreetMap[20] (OSM) — non-profit web-mapping[21] project with a detailed free-of-charge geographical map of the world.

The RoadInfoService(see page 40) of this system calls the OSM service for information about roads or separate road sections:

- Locality tag;
- Speed limits;
- Street name;
- Safety Objects;
- other parameters.

20 https://wiki.openstreetmap.org/wiki/Main_Page
21 https://ru.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D0%BA%D0%B0%D1%80%D1%82%D0%BE%D0%B3%D1%80%D0%B0%D1%84%D0%B8%D1%8F

# 4 Mobile solutions

Mobile Open source solution application is designed to introduce the proposed software product, which allows to evaluate driving behaviour of a mobile application user, to a wide range of developers and users. The mobile application has implementations for Android and iOS.

The materials of this Section:

- demonstrate the order of mobile application SDK initialisation for Android(see page 48) and iOS(see page 67);
- give examples of software product's mobile and server parts interaction(see page 103);
- describe the procedures for determining vehicle driving start and finish(see page 121);
- describe the procedure for mobile application user interface localisation (see page 133)into different languages (default language - English).

Requirements to user client devices (smartphones):

- operation system:
    - Android version 8.1 or higher;
    - iOS version 14 or higher;
- Open Source SDK version from 27 to 30;
- required permanent access permissions:
    - geolocation service
    - Motion & Fitness activity
    - Bluetooth
    - Internet
    - Application operation in the background
- disabling power-saving mode during mobile application operation.

## 4.1 Android

Open source solution SDK for Android description consists of the following components:

- Steps to start an Open source project;(see page 48)
- Steps for integrating the Open source solution into your solutions;(see page 50)
- Onboarding and main screens for Android(see page 53);
- Description of interaction with the Open source solution APIs to obtain scoring data.(see page 64)

### 4.1.1 Get started Android

For acquaintance with the Open source solution SDK you need to receive user account data and parameters for accessing repositories of source codes. For this you should:

1. Contact our Technical Support Service[22] to receive the following access parameters:

- USER - account username;
- PASSWORD - user account password;
- NAVIGATOR_URL - navigation service address URL.

22 http://r-telematica.atlassian.net/wiki/spaces/OP/pages/2395766794/Technical+support+service

2. To work with the Open source project you need to receive the access to our repositories on Github[23] that contain source codes and technical documentation of the project.

Provide our Technical Support Service with the information about your Github[24] user account - Github user e-mail address.

3. Find the sdk-telemetry/keystore.properties[25] file in the form given below in the source code repository:

```
WEB_API="https://scoring-api.kasko2go.net/api/"
RECEIVER="receiver.kasko2go.net"
```

4. Add the lines with access parameters received from out Technical Support Service to the end of the sdk-telemetry/keystore.properties[26] file on Step 1.

sdk-telemetry/keystore.properties[27] file takes the following form:

```
WEB_API="https://scoring-api.kasko2go.net/api/"
RECEIVER="receiver.kasko2go.net"

USER="<user_name>"
PASSWORD="<user_password>"
NAVIGATOR_URL="<https://navigation_url.host.com/>"
```

5. To work with Google autocomplete service receive an **API key for the Maps SDK for Android**. The procedure for receiving the key is described in Using API Keys[28].

Specify the received Google **API key for the Maps SDK for Android** in sdk-telemetry/keystore.properties[29] file, which takes the following form:

```
WEB_API="https://scoring-api.kasko2go.net/api/"
RECEIVER="receiver.kasko2go.net"

GOOGLE_API_KEY="<api_key>"
USER="<user_name>"
PASSWORD="<user_password>"
NAVIGATOR_URL="<https://navigation_url.host.com/>"
```

23 http://github.com
24 http://github.com
25 https://bitbucket.org/r-telematica/android.opensource/src/master/
26 https://bitbucket.org/r-telematica/android.opensource/src/master/
27 https://bitbucket.org/r-telematica/android.opensource/src/master/
28 https://developers.google.com/maps/documentation/android-sdk/get-api-key
29 https://bitbucket.org/r-telematica/android.opensource/src/master/

6. To work with Google maps services receive a Google **Maps API key**. The procedure for receiving the key is described in Maps SDK for Android Quickstart[30].

Specify the received Google Maps API key in **app/src/main/AndroidManifest.xml** file in the following form:

```
<meta-data
    android:name="com.google.android.geo.API_KEY"
    android:value="<key>" />
```

7. Provide interaction of your Android application with Firebase services. For this add the **google-services.json** configuration file to your mobile application. Follow this link[31] for the procedure of adding the **google-services.json** configuration file to your mobile application.

### 4.1.2 Initialization Android SDK

To integrate the Open source solution SDK into your new project, you need to perform the following steps after contacting our Technical Support:

**1. Add necessary permissions**

Add necessary permissions to your new project and ask a smartphone user to allow these permissions:

- Internet
- application operation in the background
- geolocation service
- physical activity
- disabling power-saving mode
- enabling Bluetooth adapter
- you must declare a dependency to the **Google Location and Activity Recognition** API version 12.0.0 or higher

```
    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="android.permission.ACCESS_BACKGROUND_LOCATION" />
    <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
    <uses-permission android:name="android.permission.ACTIVITY_RECOGNITION" />
    <uses-permission android:name="android.permission.REQUEST_IGNORE_BATTERY_OPTIMIZATIONS" />
    <uses-permission android:name="android.permission.BLUETOOTH" />
    <uses-permission android:name="com.google.android.gms.permission.ACTIVITY_RECOGNITION" />
```

**2. Implement metadata**

Metadata contains extra information about a user, a device and a vehicle:

- userId - unique identifier of the user
- deviceId - unique identifier of the device
- vehicleId - unique identifier of the vehicle

30 https://developers.google.com/maps/documentation/android-sdk/start
31 https://firebase.google.com/docs/android/setup#add-config-file

These identifiers allow to uniquely identify a user, are randomly generated by the mobile application, must be unique and have a constant value for each user and user device. If the mobile application has been reinstalled, these identifiers are generated again.

The identifiers userId, deviceId, vehicleId are described by the following class:

```
class Metadata(
        userId: String,
        vehicleId: String,
        deviceId: String,
        tag: String,
        val appId: String,
        val appVersion: String,
        val mobileModel: String,
        val osVersion: String,
        val isBluetoothOn: Boolean
    )
```

**3. Add speed configs**

To minimise battery power consumption, in SDK you can configure parameters of the autostart_gps_filters filter that has the following set of parameters:

- time - periodicity of obtaining coordinates from the GPS system, time in seconds
- distance - periodicity of obtaining coordinates from the GPS system, distance travelled in metres
- uspeed - upper threshold of vehicle speed, km/h
- dspeed - lower threshold of vehicle speed, km / h

The parameters of the autostart_gps_filters filter determine the periodicity with which the mobile application receives and sends GPS system data to the server for further processing. The autostart_gps_filters filter starts working after trip start validation and has the logic:

- if the current GPS speed >= dspeed and GPS speed < uspeed and the previous filtering state differs from the current one, then apply a new filter by time = time (in seconds) and filter by distance = distance (in meters).
- If the previous speed value was within the same limits as the current one, then do not change the filter characteristics.

The autostart_gps_filters filter with default parameter values is a variable with JSON of the type:

```
val configs = """
    [
    {
        "time": 18,
        "distance": 200,
        "uspeed": 70,
        "dspeed": 0
    },
    {
        "time": 18,
        "distance": 400,
        "uspeed": 110,
        "dspeed": 60
    },
    {
        "time": 18,
```

```
        "distance": 600,
        "uspeed": 150,
        "dspeed": 100
    },
    {
        "time": 18,
        "distance": 800,
        "uspeed": 190,
        "dspeed": 140
    },
    {
        "time": 18,
        "distance": 1000,
        "uspeed": 230,
        "dspeed": 180
    },
    {
        "time": 18,
        "distance": 1200,
        "uspeed": 1000,
        "dspeed": 220
    }
]

    """
    DetectorSdk.setSpeedConfig(context, configs)
```

#### 4. Working with Bluetooth module (optional)

SDK provides an opportunity to identify the trip start moment through connecting the smartphone bluetooth module to the vehicle bluetooth.

To work with the smartphone bluetooth module, follow these steps:

```
val prefs = context.getSharedPreferences(BTConnectedMonitor.BT_PREFS, Context.MODE_PRIVATE)
prefs.edit(commit = true) {
    putString(BTConnectedMonitor.BT_ADDRESS, address)
    putString(BTConnectedMonitor.BT_NAME, name)
}
```

#### 5. SDK initialisation

To initialise the SDK, follow these steps:

```
val config = ConfigBuilder(
        host = BuildConfig.RECEIVER,
        port = BuildConfig.RECEIVER_PORT
    )
        .setStartValidationTimeout(180)
        .setStopValidationTimeout(300)
        .build()
```

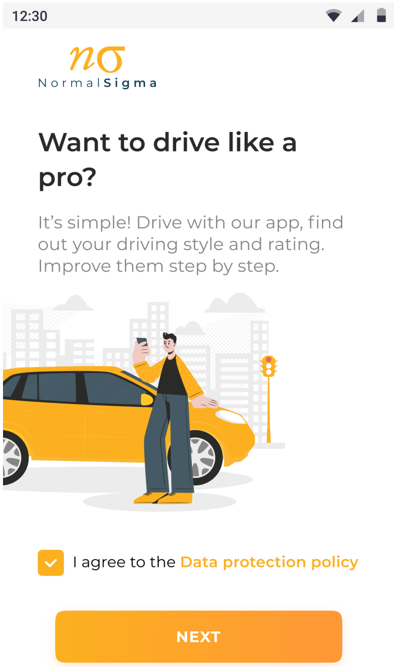
```
    DetectorSdk.start(context, config, getMetadata())
```

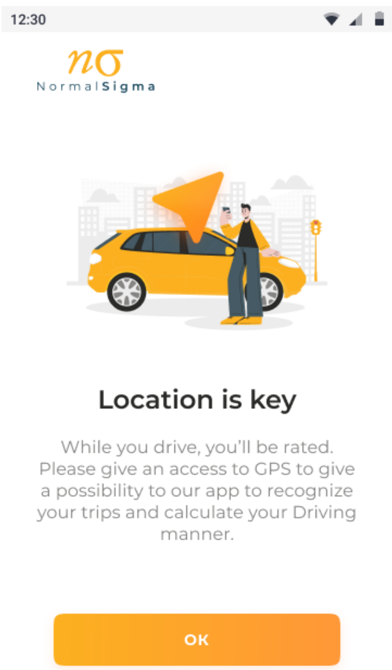
### 4.1.3 Onboarding & main screens for Android

When application is started for the first time, user should go through onboarding screens starting from 0.0 to 1.6. After that Main screen should be opened and it should be opened automatically every time when the app is started, user should go through onboarding screens starting from 2.0 to 4.1. Authorization/registration is not required.

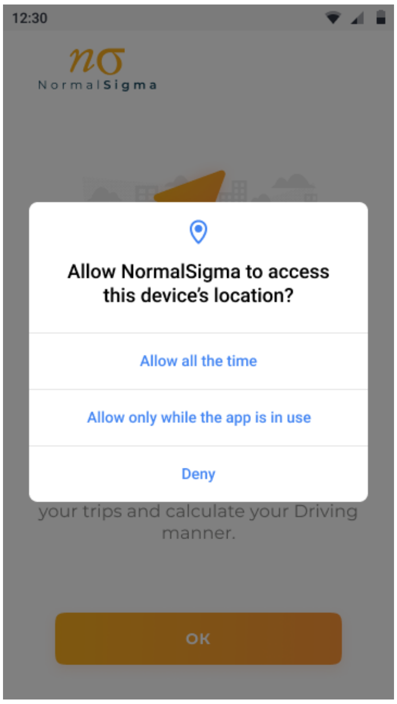
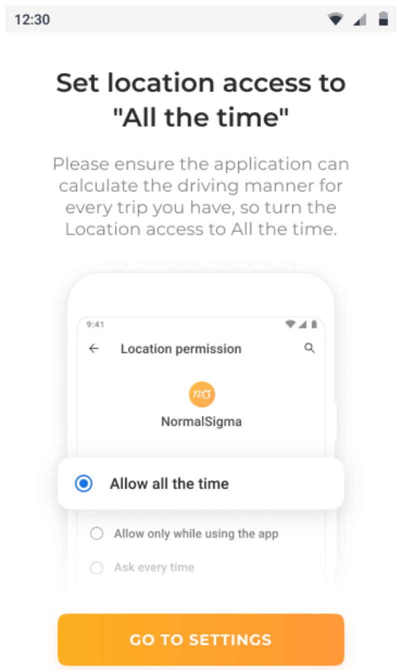
Onboarding (intro & initial settings) screens

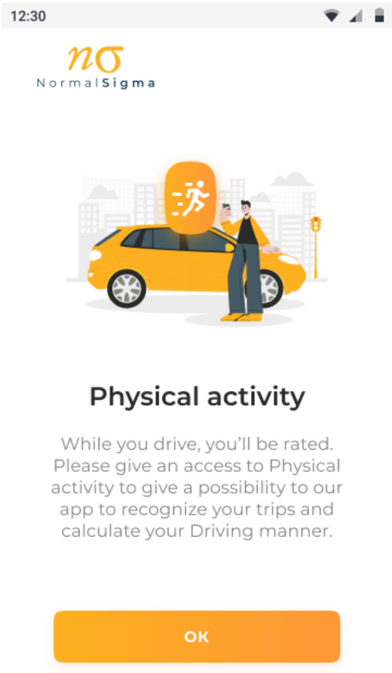
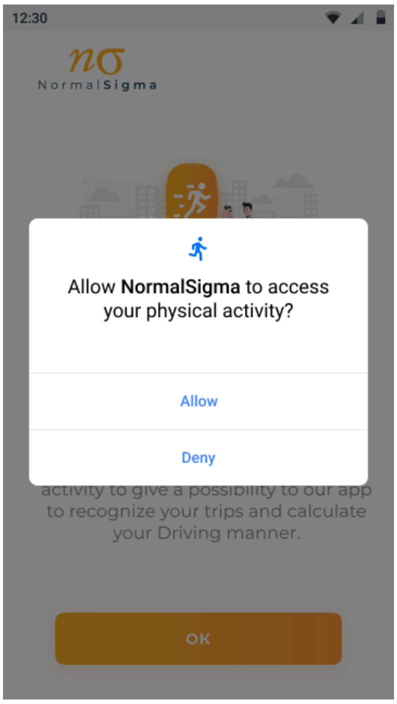
| Screen | Description | Message keys |
|---|---|---|
| 0.0 Intro<br> | Welcome screen | |

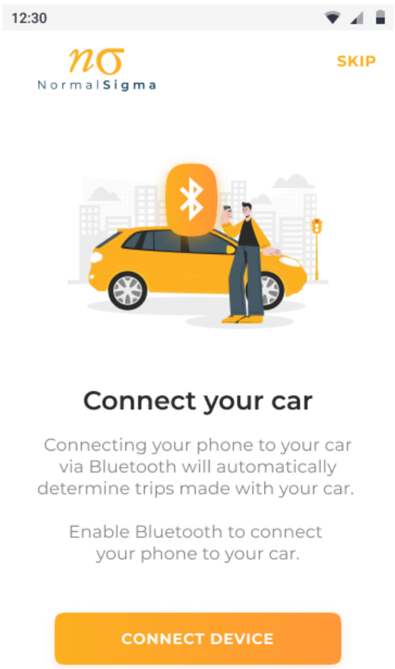
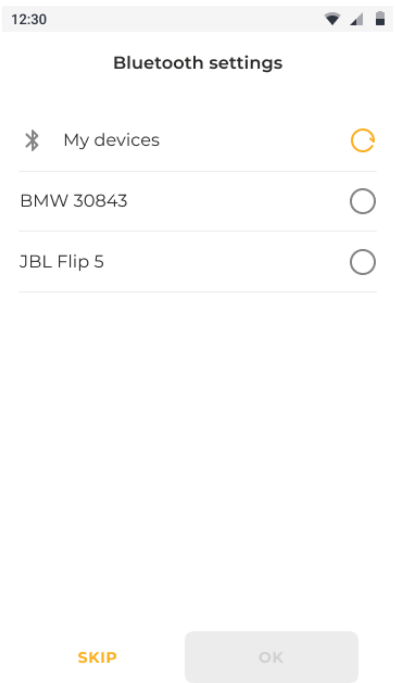
<table>
<tr><th>Screen</th><th>Description</th><th>Message keys</th></tr>
<tr>
<td>


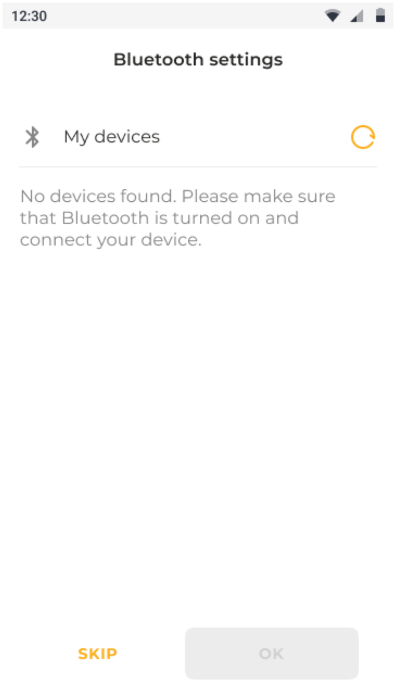
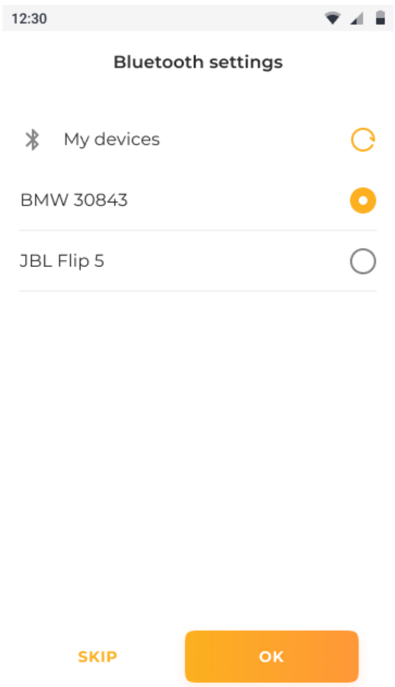
</td>
<td>Check-box:<br>1. By default is False<br>2. Must be manually set to True to activate “Next“ button and unlock further steps</td>
<td>onboard_label_header<br>onboard_label_text<br>onboard_label_privacy_begin<br>onboard_label_privacy_end<br>onboard_url_privacy<br>onboard_btn_next</td>
</tr>
<tr>
<td>1.0 “Location is key”


</td>
<td>Ok button on this screen triggers Location pop-up</td>
<td>intro_label_header<br>intro_label_text<br>intro_btn_ok</td>
</tr>
</table>

| Screen | Description | Message keys |
|---|---|---|
| 1.1 Location pop-up<br> | If a user has not provided access to sensor, then he is able to proceed to the next step anyway. In other words, next screen is opened after pop-up in any case. | system_label_gps_access |
| 1.1a Location “Allow all the time“<br> | On this screen a user is only able to go to system settings. When button “Go to settings“ is being pressed:<br>1. System settings (OS) window is opened<br>2. In background application switches to the next screen (1.2 Physical activity) after pressing “Go to settings“ button, so that user is potentially able to ignore system settings and just close system settings window and proceed with onboarding in the app | autostart_label_location_always_header<br>autostart_label_location_always_text<br>autostart_btn_go2settings |

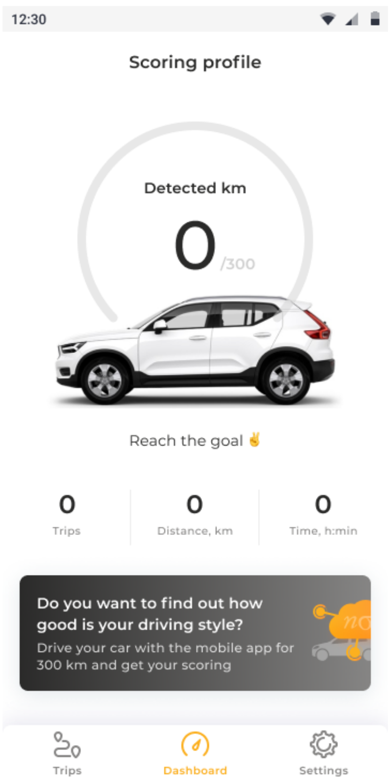
| Screen | Description | Message keys |
|---|---|---|
| 1.2 Motion and Fitness / Physical activity   | Ok button on this screen triggers 1.3 Motion pop-up | autostart_label_fitness_header_Android<br>autostart_label_fitness_text_Android |
| 1.3 Motion pop-up   | If a user has not provided access to sensor, then he is able to proceed to next step anyway. In other words, next screen is opened after pop-up in any case. | label_system_fitness_access |

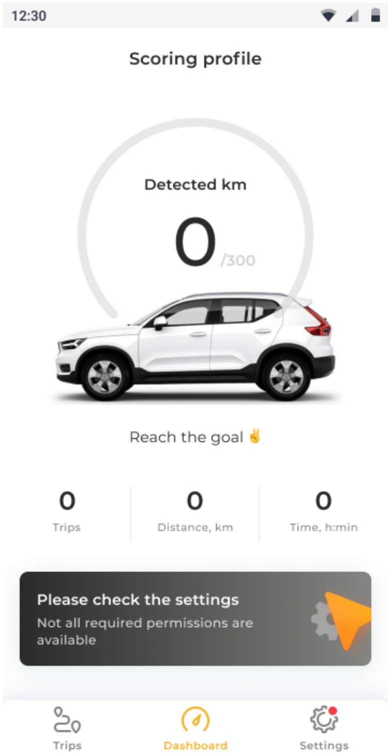
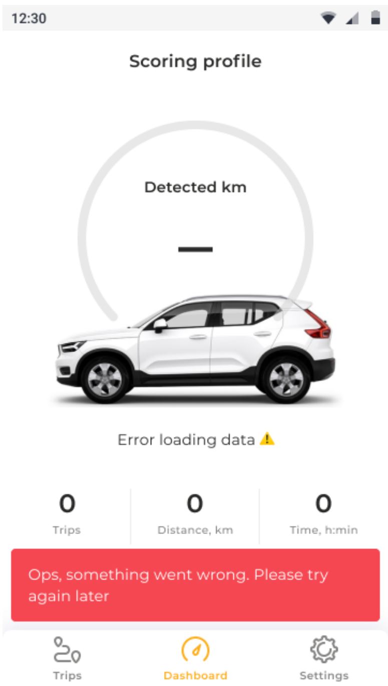
| Screen | Description | Message keys |
|---|---|---|
| 1.4 BT intro<br><br> | “Skip” button → 2.0 Main<br>“Connect device“ button → 1.5 BT settings | autostart_label_access2bluetooth_title<br>autostart_label_access2bluetooth_body<br>autostart_btn_access2bluetooth_ok<br>autostart_btn_skip |
| 1.5-1.6 BT settings<br><br> | The following scenarios are possible at this step:<br>1. Ask user to choose one of the devices that are already connected to the smpartphone<br>2. Ask user to turn on BT, attach device, refresh list of the devices and choose it for autostart | autostart_label_title<br>autostart_label_mydevices<br>autostart_label_nodevicefound<br>autostart_btn_skip<br>autostart_btn_ok |

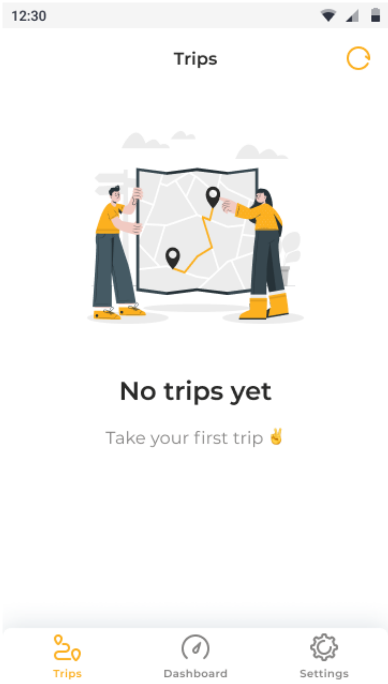
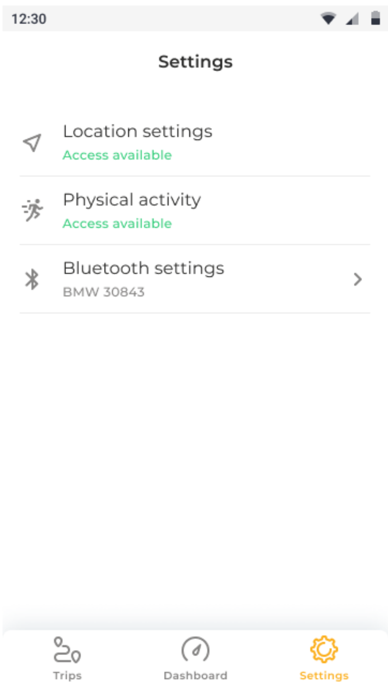
| Screen | Description | Message keys |
|---|---|---|
| 1.5a BT Settings<br> | User can proceed to 2.0 Main by pressing “Skip“ | |
| 1.6 BT Settings<br> | Usercan proceed to 2.0 Main by choosing one of the BT devices and pressing OK | |

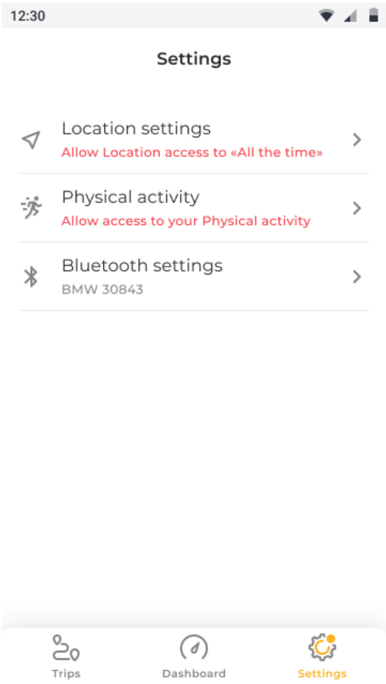
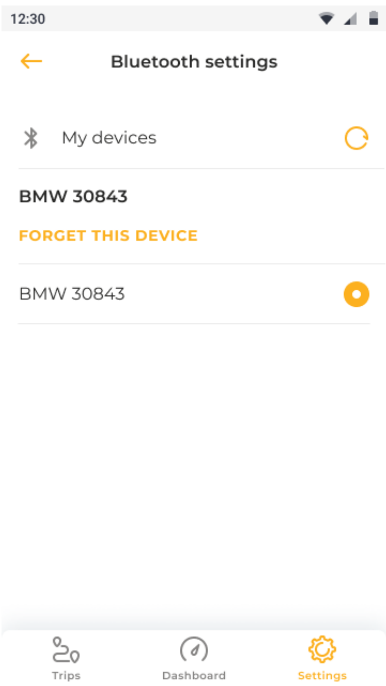
When Main screens are opened for the first time on this device, application should generate a new random vehicleGuid which should be used for subsequent scoring requests.

Main screens

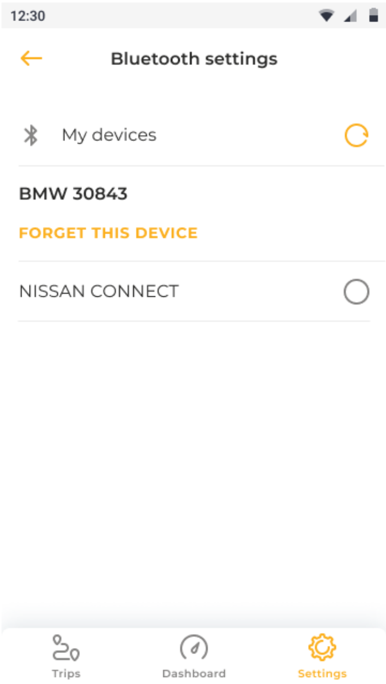
| Screen | Description | Localize |
|---|---|---|
| 2.0 Main<br> | Every time when Main screen is opened, application should retrieve scoring from server (Request: GET / api/Scoring with tag="driver_in_my_car" and response (scorePercent)) determines that the Main screen may have one of the following states:<br>2.0 No trips (tripsCount=0)<br>2.1 - At least one trip, best driving manner (scorePercent: 80-100)<br>2.1a - At least one trip, middle level of driving manner (scorePercent: 60-79)<br>2.1b - At least one trip, wrost driving manner (scorePercent: 0-59) | main_label_header<br>main_label_scoring<br>main_label_motivate (trips>0, Best manner - 2.1)<br>main_label_motivate_middle (trips>0, manner: Middle level - 2.1a)<br>main_label_motivate_wrost (trips>0, Wrost manner - 2.1b)<br>main_label_trips<br>main_label_distance<br>main_label_time<br>main_label_dashboard<br>main_label_settings<br>main_label_no_trips (2.0) |

| Screen | Description | Localize |
|---|---|---|
| 2.0a Main banner (warning)<br><br> | When Main screen is opened, application should check that the permissions settings are OK. If not, then there is an additional screen state for this situation which may overlap with the states listed above - “Incorrect settings“ (2.0a):<br>• Banner ”Check the settings” should be displayed. It should be clickable and Settings window should be opened when this banner is pressed.<br>• Red dot should be displayed on settings icon<br>This state has a higher priority than any other states. | main_label_warning_header<br>main_label_warning_text |
| 2.0b Main - Error loading data<br><br> | | main_label_connecterror<br>main_label_error |

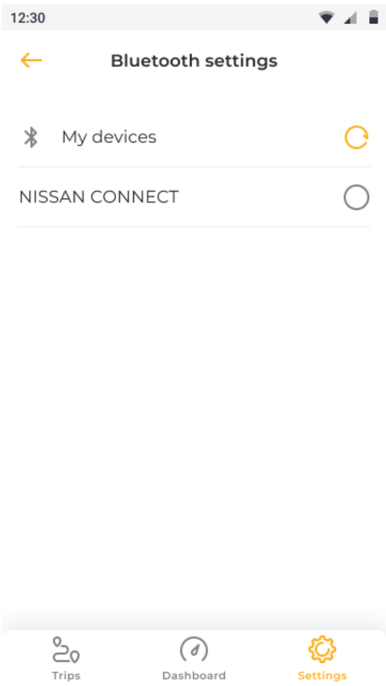
| Screen | Description | Localize |
|---|---|---|
| 3.0a No trips<br><br> | | trip_label_no_trip<br>trip_motivate_other_text |
| 4.0 Settings<br><br> | There are 3 subsections on the Settings screen:<br>1. Location settings:<br>a. If permissions are OK, then this is just a text<br>b. If permissions are not OK, then warning is displayed (4.0a) and this is a clickable area, which opens OS Settings<br>2. Fitnes/Physical activity settings - similar to Location<br>3. BT Settings - clickable subsection in which currently selected BT device is displayed and which opens 4.1 BT settings window | main_label_settings<br>settings_label_location<br>settings_label_warning_location<br>======================<br>Physical activity/Fitness:<br>autostart_label_fitness_header_Android<br>settings_label_warning_fitness_Android<br>======================<br>BT section: autostart_label_title<br>Access Ok (for Location and Fitness): settings_label_status_ok |

| Screen | Description | Localize |
|---|---|---|
| 4.0a Settings<br><br> | Permissions settings are not OK | |
| 4.1 BT Settings<br><br> | It is similar to the screen 1.5 in onboarding. In addition to the features implemented in onboarding (selection of devices, refresh list), there should be one more function: “Forget (ignore) this device” which means that a connection with a previously selected device should not trigger a trip anymore. | autostart_label_forgetdevice |

<table>
<tr><th>Screen</th><th>Description</th><th>Localize</th></tr>
<tr><td>4.1a BT Settings



</td><td></td><td></td></tr>
<tr><td>4.1b BT Settings



</td><td></td><td></td></tr>
</table>

### 4.1.4 Sample App. Android

The Open source solution SDK for Android allows a user to get the following information:

**1. Data on trips made over a specified period of time.**

Use the following API method (see page 103) to receive information on trips made over a specified period of time:

```
ScoringSdk.getTrips(
            vehicleUUID = getVehicleId(),
            deviceId = getDeviceId(),
            userId = getUserId(),
            to = DateTime.now().millis / 1000,
            from = null
    )

    fun getTrips(
        vehicleUUID: UUID, // vehicle id for identification
        deviceId: UUID, // device id for identification
        userId: UUID, // user id for identification
        to: Long?, // date to filed for filtering
        from: Long?, // date from filed for filtering
        limit: Int = 20, // limit of records in response for pagination
        offset: Int = 0 // offset of records for pagination
    ): Single<ScoringTrips>

    data class ScoringTrips(
        val limit: Long?, // limit of records in response for pagination
        val totalCount: Long, // total count of records in response for pagination
        val values: List<ScoringTrip>
    )

    data class ScoringTrip(
        val tripData: TripData,
        val scoring: ScoreData?
    )

    data class TripData(
        val id: UUID, // Unique identifier of the trip
        val userId: String?, // Unique identifier of the user
        val deviceId: String?, // Unique identifier of the device
        val vehicleId: String?, // Unique identifier of the vehicle
        val startTimestamp: DateTime, // Start time of the trip (UNIX timestamp)
        val finishTimestamp: DateTime, // Finish time of the trip (UNIX timestamp)
        val tag: TripDataTag // A value that the trip has been tagged with
        val isBluetoothOn: true, // Is bluetooth on
        val isBluetoothConnectionEstablished: true // Is bluetooth connection establihed
    )
```

**2. Detailed information on the trip that has been made.**

Use the following API method(see page 110) to receive information on a specified trip:

```
ScoringSdk.getTripsInfo(tripId)

    fun getTripsInfo(
        tripId: UUID, // id of trip
        loadAllEvents: Boolean = true, // flag for load all events
        mapToRoads: Boolean = true // flag for mapping
    ): Single<ScoringTripDetails>

    data class ScoringTripDetails(
        val trip: ScoredTrip, // trip details
        val events: List<TelemetryEvent>? = null, // trip events
        val penalties: List<Penalty>, // trip penalties
        val startAddress: String, // start trip address
        val endAddress: String, // end trip address
        val interactiveMapEnabled: Boolean? // ?
    )

    data class TelemetryEvent(
        val timestamp: Long, // Date and time of the event (UNIX timestamp)
        val latitude: Double, // Latitude of the location point (in signed degrees format)
        val longitude: Double, // Longitude of the location point (in signed degrees format)
        val speedKph: Double, // The speed of the object at the specified time (in kilometers per hour)
        val heading: Double, // ompass direction in which the object's bow or nose is pointed (0 or 360
indicates a direction toward true North)
        val accuracy: Double // The accuracy of the location information
    )

    data class Penalty(
        val timestamp: Long, // Date and time of the event (UNIX timestamp)
        val type: PenaltyType, // The type of the event
        val durationMs: Long, // The duration (in milliseconds) of the event
        val value: Double // Indicates the severity of the event (depends on the type)
    )

    enum class PenaltyType {
        BRAKING,
        ACCELERATION,
        OVER_SPEED,
    }
```

**3. The weighted average of the score and the total value of vehicle mileage over a certain period of time.**

Use the following API method(see page 116) to receive information:

```
ScoringSdk.getScoring(
            vehicleUUID = getVehicleId(),
            deviceId = getDeviceId(),
            userId = getUserId(),
            dateFrom = 0
    )

    fun getScoring(
```

```
        vehicleUUID: UUID, // vehicle id for identification
        deviceId: UUID, // device id for identification
        userId: UUID, // user id for identification
        dateFrom: Long,  // date from filed for filtering
        tag: TripDataTag = TripDataTag.DRIVER_IN_MY_CAR // tag field for filtering
    ): Single<ScoreData>

    data class ScoreData(
        val scorePercent: Double, // score percent for all selected trips
        val durationSec: Long, // duration in sec for all selected trips
        val distanceMeters: Long, // distance in meters for all selected trips
        val tripsCount: Long // count of for all selected trips
    )
```

**4. Building a vehicle route/routes, getting information about dangerous road sections.**

Use the following API method to get the information about vehicle routes and risks related to traveling along each of those routes:

Request method:

```
RoutesSdk.getRoutes(
        originLat: Double,
        originLng: Double,
        destinationLat: Double,
        destinationLng: Double,
        departureTime: Long?,
)
```

Response parameters:

```
@Parcelize
data class Route(
    val distance: Double,
    val duration: Long,
    val accidentRisk: Double,
    val riskCountForUi: Int,
    val riskCountForUiColor: String,
    val inactiveRouteColor: String,
    val lowRiskDistance: Double,
    val lowRiskPercentage: Double,
    val normalRiskDistance: Double,
    val normalRiskPercentage: Double,
    val highRiskDistance: Double,
    val highRiskPercentage: Double,
    val highRiskLinks: List<HighRiskLink>,
    val road: Road,
    val waypoints: List<LatLng>
) : Parcelable

@Parcelize
```

```
data class HighRiskLink(
    val description: String,
    val accidentsCount: Int,
    val accidentsYears: String,
) : Parcelable

@Parcelize
data class Road(
    val baseRoute: List<LatLng>,
    val baseColor: String,
    val lowRoute: List<List<LatLng>>,
    val lowColor: String,
    val normalRoute: List<List<LatLng>>,
    val normalColor: String,
    val highRoute: List<List<LatLng>>,
    val highColor: String,
) : Parcelable

@Parcelize
data class RoutesWrapper(
    val routes: List<Route>
) : Parcelable
```

Link to Swagger:

- paragraphs 1-3 - https://scoring-api.kasko2go.net/swagger/index.html
- paragraph 4 - http://accidentprobabilitycalculator-service/swagger/index.html

## 4.2 iOS

Open source solution SDK for iOS description consists of the following components:

- Steps to start an Open source project;(see page 67)
- Steps for integrating the Open source solution into your solutions;(see page 68)
- Onboarding and main screens for iOS;(see page 73)
- Description of interaction with the Open source solution APIs to obtain scoring data .(see page 94)

### 4.2.1 Get started iOS

To try the Open source solution SDK you need to receive registration data, parameters of access to source code repositories and our network infrastructure.

Contact our Technical Support[32] to get the following information:

| Name | Description |
|---|---|
| mvnUrl | Repository URL |

32 http://r-telematica.atlassian.net/wiki/spaces/OP/pages/2395766794/Technical+support+service

| **Name** | **Description** |
|---|---|
| mvnUserName | User name for repository access |
| mvnPassword | Access password for repository |
| serverAddress | server IP address |
| serverPort | server port number |

### 4.2.2 Initialization iOS SDK

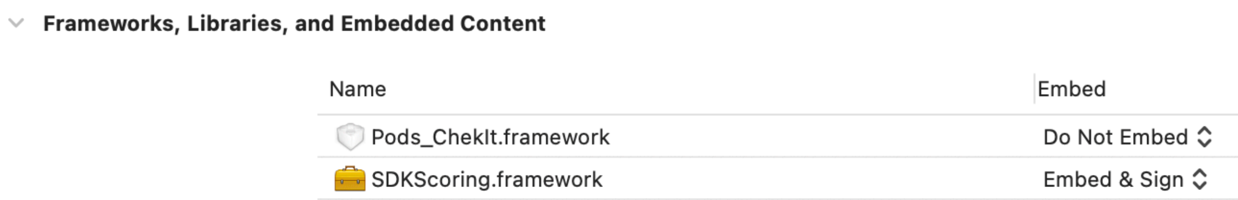
To integrate the Open source solution SDK into your new project, you need to perform the following steps after receiving primary information in our Technical Support:

**1. Add framework to project**

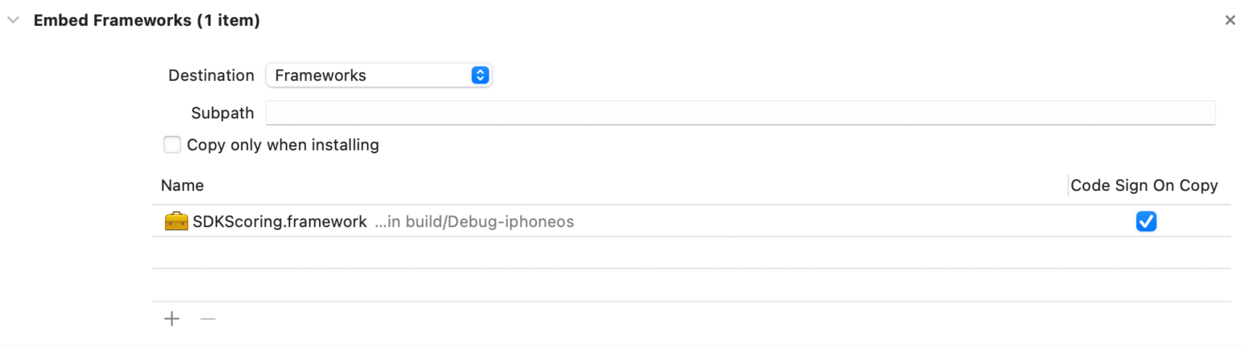
Download from the repository and add to your new project, for example into IDE Xcode, our framework **SDKScoring.framework**; if needed, set the flag “Copy items if needed“.

While configuring IDE Xcode, perform the following steps:

- Set the value **Embed & Sign** in the **General** section for the framework **SDKScoring.framework**




- In the **Build Phases** section add **SDKScoring.framework** into **Embed Frameworks** subsection




- In the **Build Phases** section add **SDKScoring.framework** into **Link Binary With Libraries** subsection.

**2. Add necessary permissions**

Add necessary permissions to your new project and ask a smartphone user to allow these permissions:

- geolocation service (requestAlwaysAuthorization)
- physical activity sensor (queryActivityStarting).

To obtain the appropriate access rights, add the following strings to the settings file of your project *Info.plist*:

<key>NSLocationAlwaysAndWhenInUseUsageDescription</key>

<string>We use Location services to rate your driving</string>

<key>NSLocationAlwaysUsageDescription</key>

<string>We use Location services to rate your driving</string>

<key>NSLocationWhenInUseUsageDescription</key>

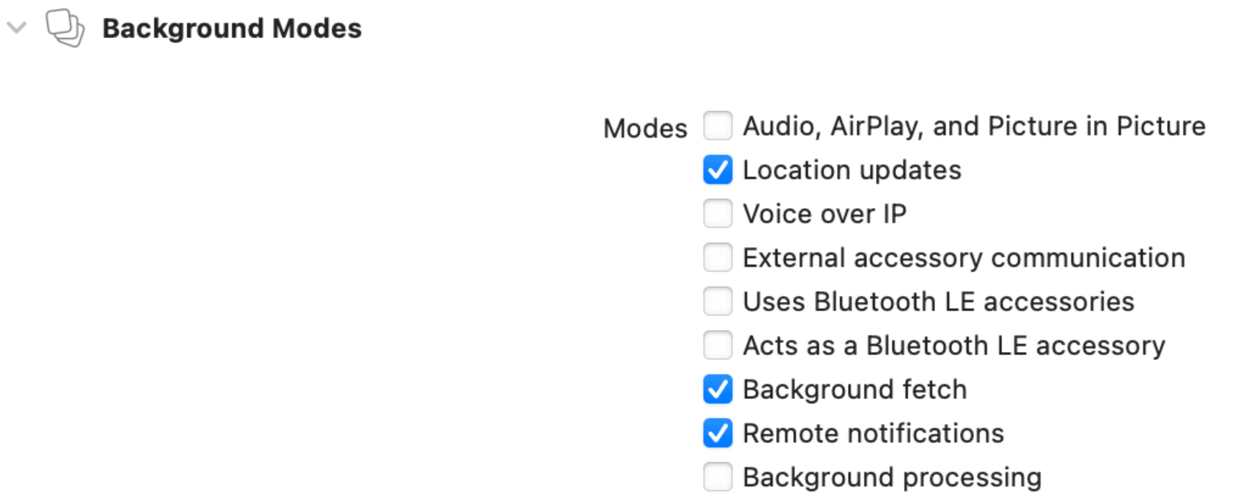
<string>We use Location services to rate your driving</string>

<key>NSMotionUsageDescription</key>

<string>We only use motion data to rate your driving.</string>

**3. Background processes**

Allow mobile application operation in the iOS background; to do this, set flags **Location updates** and **Background fetch** for **Background Modes** in the **Signing & Capabilities** section in IDE Xcode.




**4. SDK configuration**

To implement SDK into the required classes of your project perform: import SDKScoring.

Interaction with SDK should be done through: public class - ScoringUserBehaviourObserver.

SDK configuration is done using the following methods.

**Declaration:** ScoringUserBehaviourObserver.shared.setup(with userID: String, vehicleID: String, deviceID: String, isBluetoothOn: Bool, settingsArray: [[String : Any]]?, with loggingIsOn: Bool)

where:

| Name | Descriptoin |
| --- | --- |

<table>
<tr><td>userId</td><td>• unique identifier of the user</td></tr>
<tr><td>deviceId</td><td>• unique identifier of the device</td></tr>
<tr><td>vehicleId</td><td>• unique identifier of the vehicle</td></tr>
<tr><td></td><td>These identifiers allow to uniquely identify a user, are randomly generated by the mobile application, should be unique and have a constant value for each user and user device. If the mobile application has been reinstalled, these identifiers are generated again.</td></tr>
<tr><td>isBluetoothOn</td><td>• flag that indicates a bluetooth device saved by the application</td></tr>
<tr><td>settingsArray</td><td>• an array of parameters that determine the periodicity with which the mobile application receives and sends GPS system data to the server for further processing.<br>settingsArray contains the following set of parameters:
<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>time</td><td>• periodicity of receiving coordinates from the GPS system, time in seconds</td></tr>
<tr><td>distance</td><td>• periodicity of receiving coordinates from the GPS system, distance travelled in metres</td></tr>
<tr><td>uspeed</td><td>• upper vehicle speed threshold, km/h</td></tr>
<tr><td>dspeed</td><td>• lower vehicle speed threshold, km/h</td></tr>
</table>
</td></tr>
</table>

| | |
|---|---|
| | The mobile application will start receiving and sending GPS system data to the server for further processing after trip start validation(see page 121) and is described by the logic:<br>• if the current GPS speed >= dspeed and GPS speed < uspeed and the previous filtering state differs from the current one, then apply a new filter by time = time (in seconds) and filter by distance = distance (in meters).<br>• If the previous speed value was within the same limits as the current one, then do not change the filter characteristics.<br>settingsArray with default parameter values is as follows:<br>let exampleSettings: [[String:**Any**]] = [<br>[<br>"time": 18,<br>"distance": 200,<br>"uspeed": 70,<br>"dspeed": 0<br>],<br>[<br>"time": 18,<br>"distance": 400,<br>"uspeed": 110,<br>"dspeed": 60<br>],<br>[<br>"time": 18,<br>"distance": 600,<br>"uspeed": 150,<br>"dspeed": 100<br>],<br>[<br>"time": 18,<br>"distance": 800,<br>"uspeed": 190,<br>"dspeed": 140<br>],<br>[<br>"time": 18,<br>"distance": 1000,<br>"uspeed": 230,<br>"dspeed": 180<br>],<br>[<br>"time": 18,<br>"distance": 1200,<br>"uspeed": 1000,<br>"dspeed": 220<br>]] |
| loggingIsOn | • flag resposible for collecting and sending logs to the server. |

| | | |
|---|---|---|
| | Logs composition: | |
| | “TRIP_START_CANDIDATE" | • trip start candidate is detected |
| | “TRIP_START_VALID" | • trip start candidate is validated |
| | “TRIP_STOP_CANDIDATE" | • trip stop candidate is detected |
| | “TRIP_STOP_VALID" | • trip stop candidate is validated |
| | “TRIP_STOP_INVALID" | • trip start candidate is not validated |
| | “DETECTOR_SETTINGS" | • settings have been received |
| | “BT_CONNECTED" | • saved Bluetooth device is connected |
| | “BT_DISCONNECTED" | • saved Bluetooth device is disconnected |
| | “TELEMETRY_SENDER_START" | • start of telemetry data sending |
| | “AuthRequested" | • request for authorisation with server |
| | “Handshake" | • authorisation with server |
| | “TELEMETRY_SENDER_STOP" | • stop of telemetry data sending |
| | “TELEMETRY_PACKAGE_ERROR" | • error during sending telemetry data to the server |
| | “STATUS_ACTIVITY_IN_VEHICLE" | • user activity change (in vehicle/ out of vehicle) |
| | “EXIT_REGION" | • exit from the given region (by default - 100 m) |
| | “SIGNIFICANT_LOCATION_CHANGE" | • significant user location change |
| | “STATUS_GPS_MODE" | • true, if the GPS authorisation status is authorizedAlways, false otherwise |

| | | |
|---|---|---|
| | “STATUS_FITNESS" | • true, if access to the fitness sensor is allowed |
| | “STATUS_POWER_SAFE" | • true, if power-saving mode is enabled |
| | “STATUS_DISK” | • free disk space in Mb |
| | “STATUS_OK" | • general system status |

The trip detection alrogithm(see page 121) is started as follows:

**Declaration:** ScoringUserBehaviourObserver.shared.startMonitoringForRegion().

To avoid losing data collected by the mobile application in case of applicaton termination, go to *AppDelegate* of the application and in the applicationWillTerminate **method** call ScoringUserBehaviourObserver.shared.terminated() **method**:

```
func applicationWillTerminate(_ application: UIApplication) {
ScoringUserBehaviourObserver.shared.terminated()
}
```

When connecting/disconnecting a saved bluetooth device it is necessary to use the following methods:

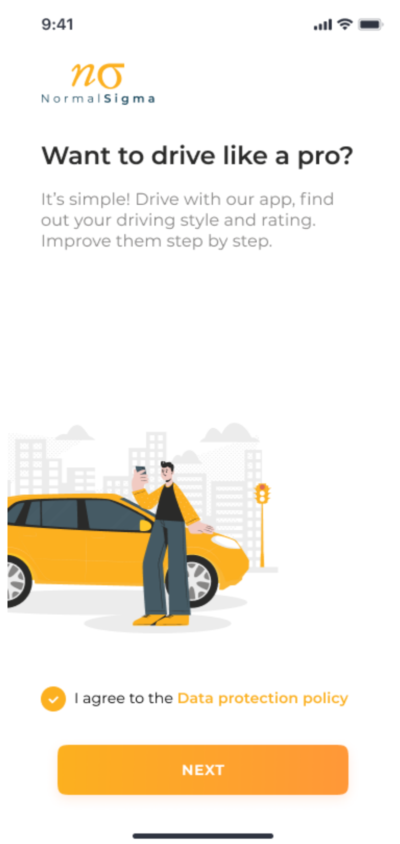
- Declaration**:** ScoringUserBehaviourObserver.shared.pairedDeviceIsActive() - when connecting a bluetooth device
- Declaration**:** ScoringUserBehaviourObserver.shared.pairedDeviceIsInactive() - when disconnecting a bluetooth device

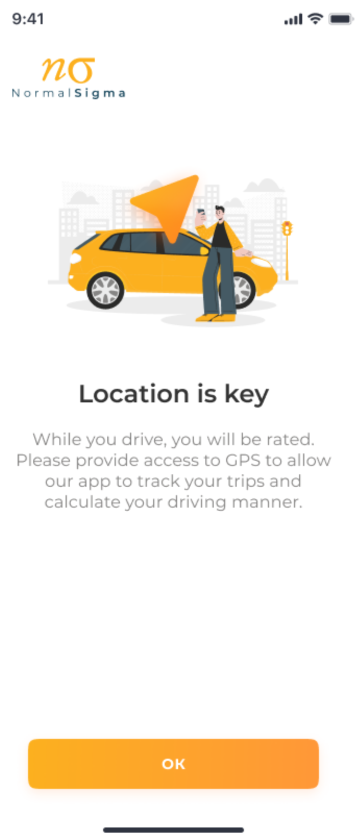
### 4.2.3 Onboarding & main screens for iOS

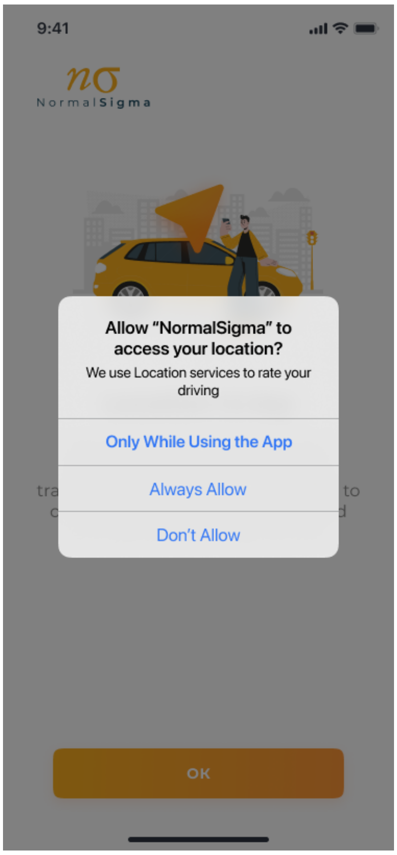
When application is started for the first time, user should go through onboarding screens starting from 0.0 to 1.8. After that Main screen should be opened and it should be opened automatically every time when the app is started, user should go through onboarding screens starting from 2.0 to 4.1. Authorization/registration is not required.

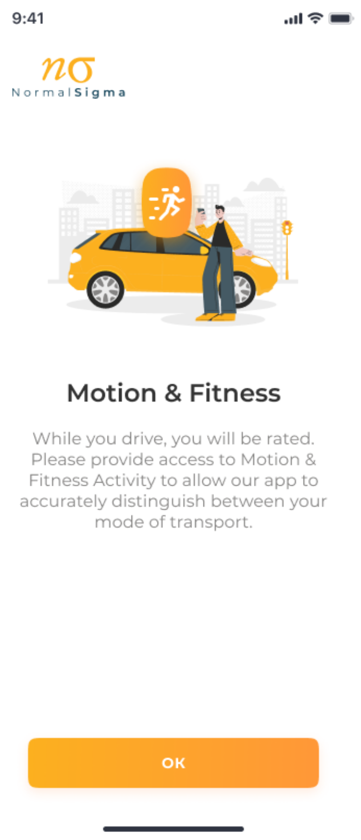
Onboarding (intro & initial settings) screens

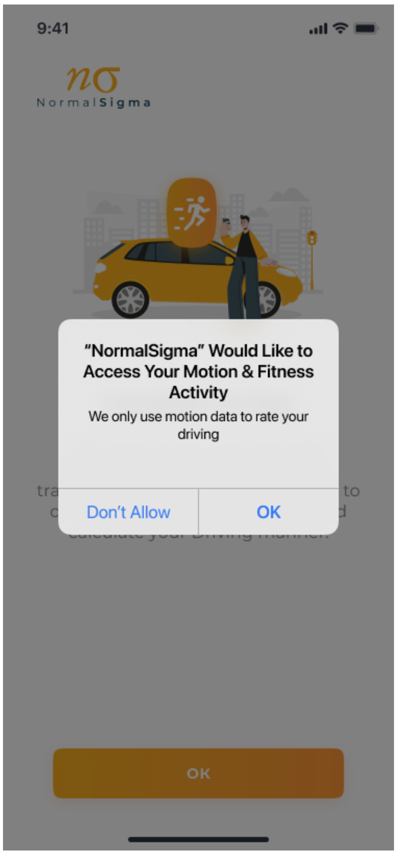
| Screen | Description | Message keys |
|---|---|---|
| 0.0 Intro   | Welcome screen | |

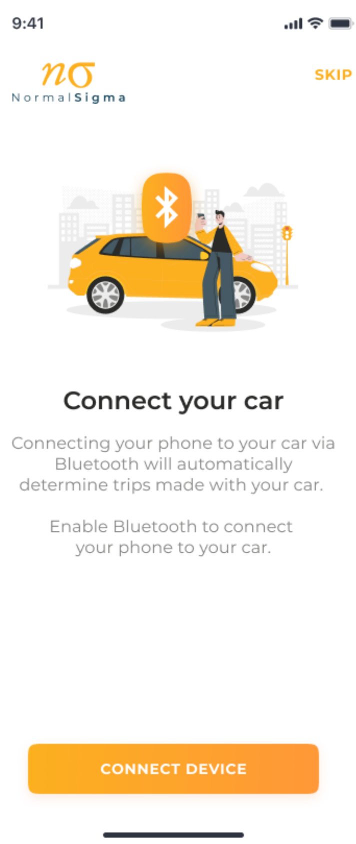
| Screen | Description | Message keys |
|---|---|---|
|  | Check-box:<br>1. By default is False<br>2. Must be manually set to True to activate "Next" button and unlock further steps | onboard_label_header<br>onboard_label_text<br>onboard_label_privacy_begin<br>onboard_label_privacy_end<br>onboard_url_privacy<br>onboard_btn_next |

| Screen | Description | Message keys |
| --- | --- | --- |
| 1.0 “Location is key”<br><br> | OK button on this screen triggers Location pop-up | intro_label_header<br>intro_label_text<br>intro_btn_ok |

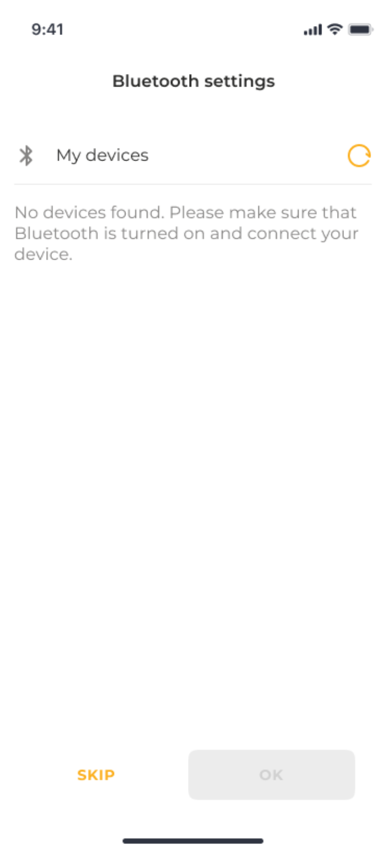
| Screen | Description | Message keys |
|---|---|---|
| 1.1 Location pop-up   | If a user has not provided access to sensor, then he is able to proceed to next step anyway. In other words, next screen is opened after pop-up in any case. | system_label_gps_access |

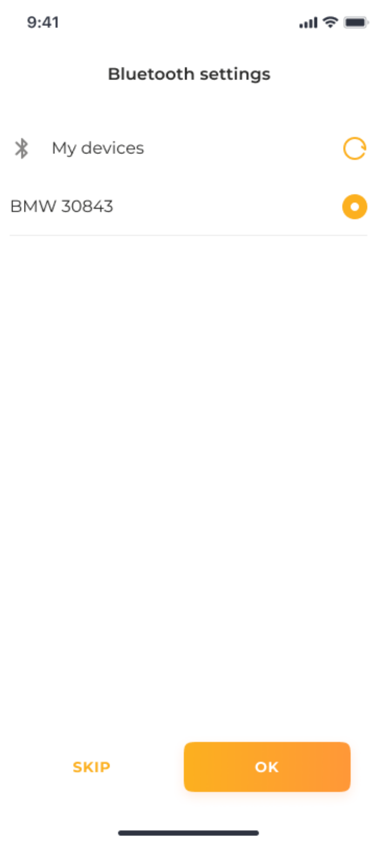
| Screen | Description | Message keys |
| --- | --- | --- |
| 1.2 Motion and Fitness / Physical activity   | OK button on this screen triggers 1.3 Motion pop-up | autostart_label_fitness_header<br>autostart_label_fitness_text |

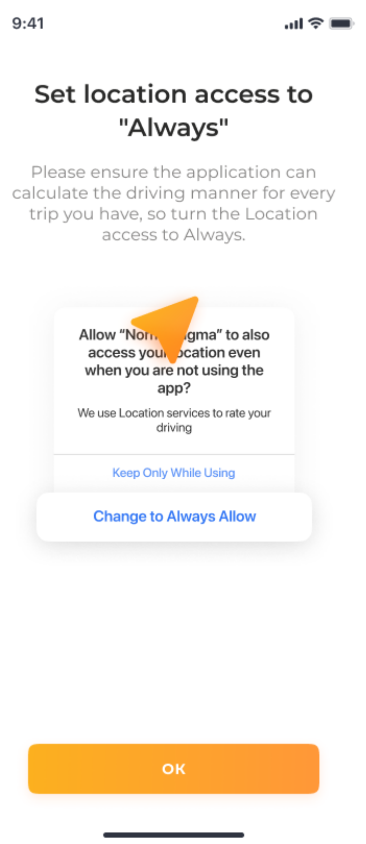
| Screen | Description | Message keys |
|---|---|---|
| 1.3 Motion pop-up   | If a user has not provided access to sensor, then he is able to proceed to next step anyway. In other words, next screen is opened after pop-up in any case. | label_system_fitness_access |

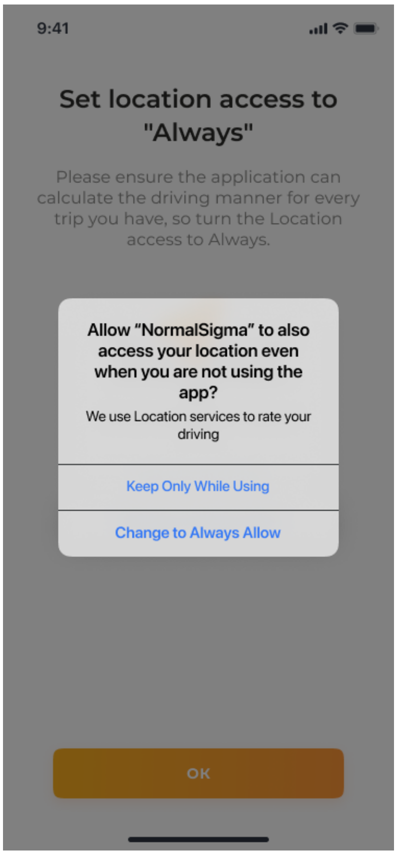
| Screen | Description | Message keys |
|---|---|---|
| 1.4 BT intro   | “Skip” button → 2.0 Main<br>“Connect device“ button → 1.5 BT settings | autostart_label_access2bluetooth_title<br>autostart_label_access2bluetooth_body<br>autostart_btn_access2bluetooth_ok<br>autostart_btn_skip |

| Screen | Description | Message keys |
|---|---|---|
| 1.5-1.6 BT settings<br><br> | The following scenarios are possible at this step:<br>1. Ask user to choose one of the devices that are already connected to the smpartphone<br>2. Ask user to turn on BT, attach device, refresh list of the devices and choose it for autostart | autostart_label_title<br>autostart_label_mydevices<br>autostart_label_nodevicefound<br>autostart_btn_skip<br>autostart_btn_ok |

| Screen | Description | Message keys |
|---|---|---|
|   | User can proceed to 2.0 Main by pressing “Skip“ | |

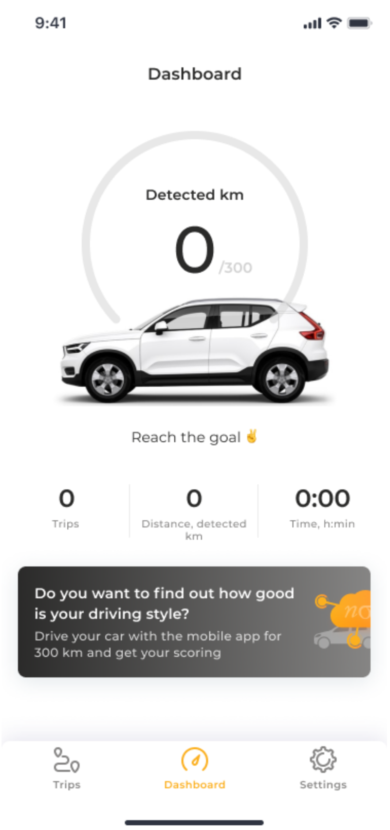
| Screen | Description | Message keys |
|---|---|---|
|   | User can proceed to 2.0 Main by choosing one of the BT devices and pressing Ok | |

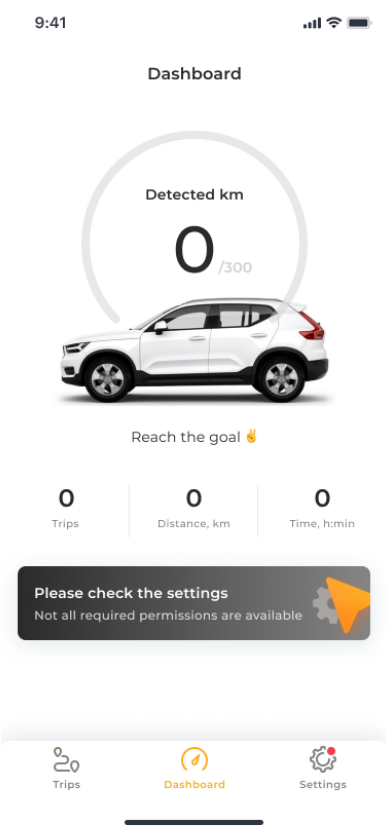
| Screen | Description | Message keys |
|---|---|---|
| 1.7 Location “Always“ info screen   | Request for Location permissions. | autostart_label_location_always_header<br>autostart_label_location_always_text_2 |

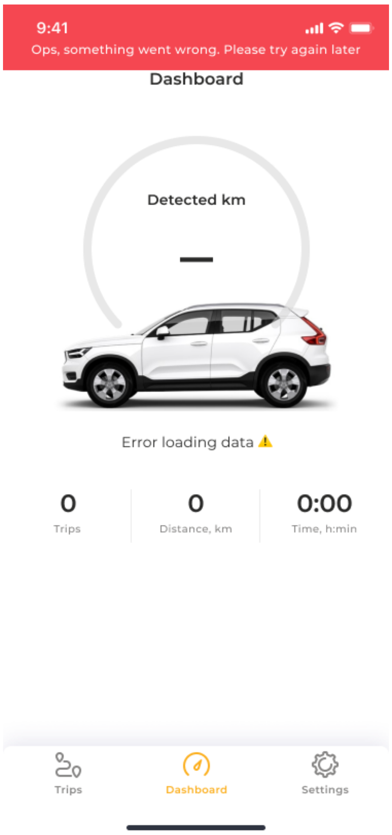
| Screen | Description | Message keys |
|---|---|---|
| 1.8 Location “Always“ pop-up   | Next screen (2.0 Main) should be opened in any case after this pop-up. | system_label_gps_access (the same as 1.1) |

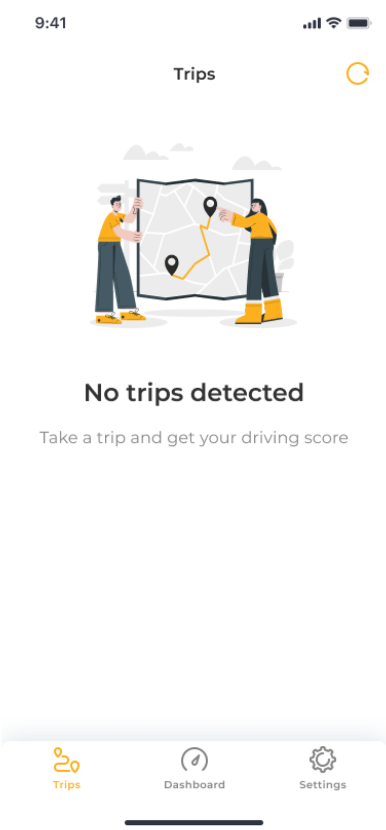
When Main screens are opened for the first time on this device, application should generate a new random vehicleGuid which should be used for subsequent scoring requests.

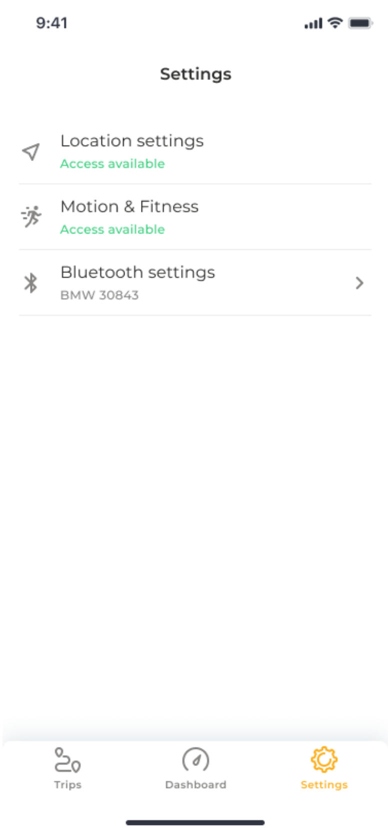
Main screens

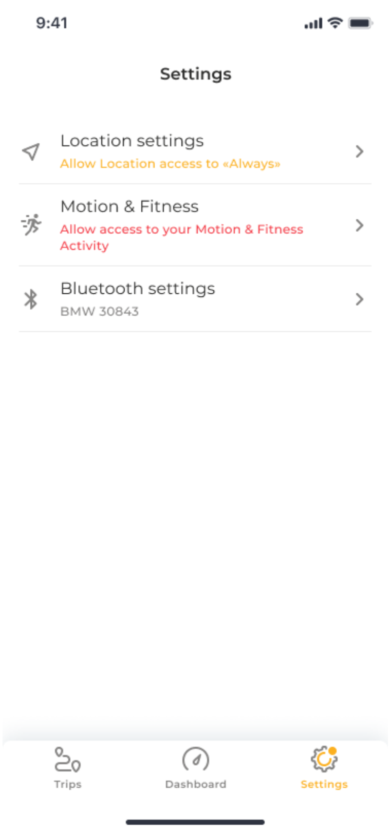
| Screen | Description | Localize |
|---|---|---|
| 2.0 Main<br> | Every time when Main screen is opened, application should retrieve scoring from server (Mobile: GET / api/Scoring with tag="driver_in_my_car") and response (scorePercent) determines that the Main screen may have one of the following states:<br>2.0 No trips (tripsCount=0)<br>2.1 - At least one trip, best driving manner (scorePercent: 80-100)<br>2.1a - At least one trip, middle level of driving manner (scorePercent: 60-79)<br>2.1b - At least one trip, wrost driving manner (scorePercent: 0-59) | main_label_header<br>main_label_scoring<br>main_label_motivate (trips>0, Best manner - 2.1)<br>main_label_motivate_middle (trips>0, manner: Middle level - 2.1a)<br>main_label_motivate_wrost (trips>0, Wrost manner - 2.1b)<br>main_label_trips<br>main_label_distance<br>main_label_time<br>main_label_dashboard<br>main_label_settings<br>main_label_no_trips (2.0) |

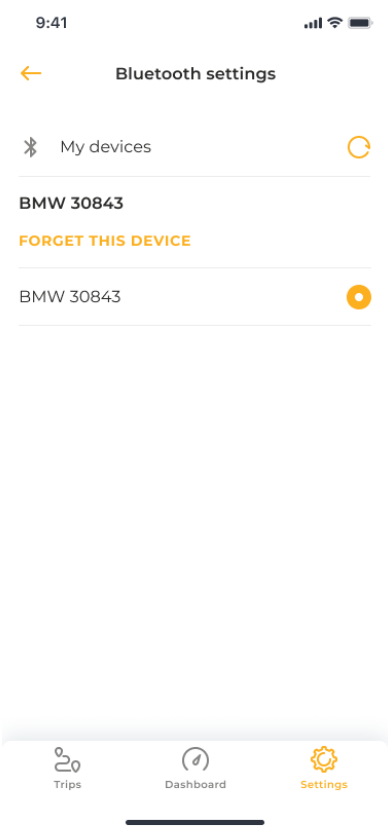
| Screen | Description | Localize |
|---|---|---|
| 2.0a Main banner (warning)<br><br> | When main screen is opened, application should check that the permissions settings are OK. If not, then there is an additional screen state for this situation which may overlap with the states listed above - “Incorrect settings“ (2.0a):<br>• Banner ”Check the settings” should be displayed. It should be clickable and Settings window should be opened when this banner is pressed.<br>• Red dot should be displayed on settings icon<br>This state has a higher priority than any other states. | main_label_warning_header<br>main_label_warning_text |

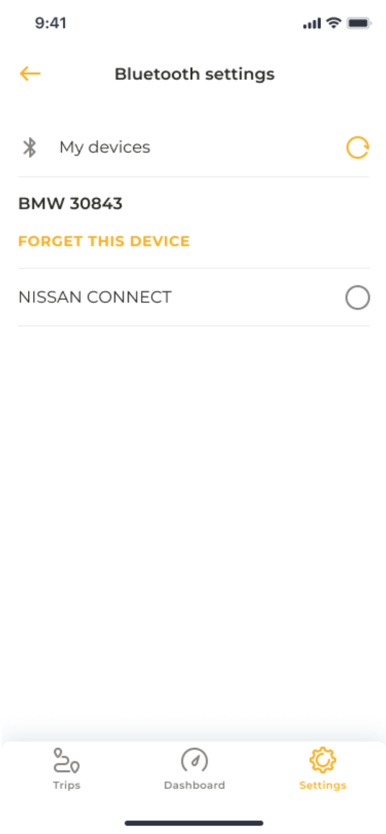
| Screen | Description | Localize |
|---|---|---|
| 2.0b Main - Error loading data   | | main_label_connecterror<br>main_label_error |

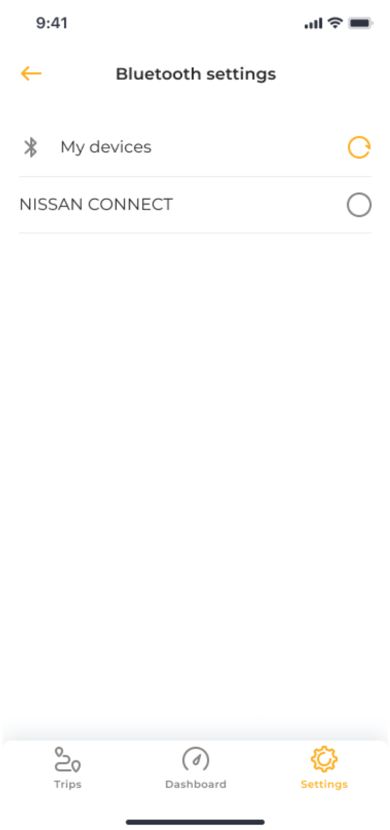
| Screen | Description | Localize |
|---|---|---|
| 3.0a No trips   | | trip_label_no_trip<br>trip_motivate_other_text |

<table>
<tr><th>Screen</th><th>Description</th><th>Localize</th></tr>
<tr>
<td>4.0 Settings<br>
</td>
<td>There are 3 subsections on the Settings screen:
<ol>
<li>Location settings:
<ol type="a">
<li>If permissions are OK, then this is just a text</li>
<li>If permissions are not OK, then warning is displayed (4.0a) and this is a clickable area, which opens OS Settings</li>
</ol></li>
<li>Fitnes/Physical activity settings - similar to Location</li>
<li>BT Settings - clickable subsection in which currently selected BT device is displayed and which opens 4.1 BT settings window</li>
</ol></td>
<td>main_label_settings<br>settings_label_location<br>settings_label_warning_location<br>=======================<br>Physical activity/Fitness:<br>iOS:<br>autostart_label_fitness_header<br>settings_label_warning_fitness<br>=======================<br>BT section: autostart_label_title<br>Access Ok (for Location and Fitness): settings_label_status_ok</td>
</tr>
</table>

| Screen | Description | Localize |
| --- | --- | --- |
| 4.0a Settings   | Permissions settings are not OK | |

| Screen | Description | Localize |
|---|---|---|
| 4.1 BT Settings   | It is similar to the screen 1.5 in onboarding. In addition to the features implemented in onboarding (selection of devices, refresh list), there should be one more function: “Forget (ignore) this device” which means that a connection with a previously selected device should not trigger a trip anymore. | autostart_label_forgetdevice |

| Screen | Description | Localize |
|---|---|---|
| 4.1a BT Settings   | | |

| Screen | Description | Localize |
| --- | --- | --- |
| 4.1b BT Settings   | | |

## 4.2.4 Sample App. iOS

The Open source solution SDK for iOS allows a user to receive the following information.

**1. Data on trips made over a certain period of time.**

Use the method

Declaration: ScoringUserBehaviourObserver.shared.getTripHistory(from beginDate: Date, to endDate: Date, tag: String, and limit: Int, with completion: @escaping ([[String : Any]]?) -> Void)

where

beginDate - date/time of the beginning of the period

endDate - date/time of the end of the period

tag(see page 20) - trip type

limit - maximum number of trips returned from the server

The method returns an array of dictionaries with the information on trips:(see page 103)

```
{
        "tripData": {
          "id": UUID, // Unique identifier of the trip
          "userId": UUID, // Unique identifier of the user
          "deviceId": UUID, // Unique identifier of the device
          "vehicleId": UUID, // Unique identifier of the vehicle
          "startTimestamp": 0, // Start time of the trip (UNIX timestamp)
          "finishTimestamp": 0, // Finish time of the trip (UNIX timestamp)
          "tag": "string", // A value that the trip has been tagged with
          "isBluetoothOn": true, // Is bluetooth on
          "isBluetoothConnectionEstablished": true // Is bluetooth connection establihed
        },
        "scoring": {
          "scorePercent": 0, // A score of the driving manner safety (100% indicates safe driving)
          "durationSec": 0, // The duration of the specified period (on seconds)
          "distanceMeters": 0, // The distance covered by the vehicle during the specified period
          "accidentness": 0,// An average accidentness score during the specified period
          "version": "string", // Versions of the components that affect the score
          "errors": "string" // Error information if any scoring component fails
        }
```

**Example:**

*<Array<Dictionary<String, Any>>>*

▽ *1 : 2 elements*

▽ 0 : 2 elements

- key : "tripData"

▽ value : 8 elements

▽ 0 : 2 elements

- key : userId
- value : 087fbef5-d257-434a-9a21-1220cfb797cf

▽ 1 : 2 elements

- key : deviceId
- value : 720c62fc-73bf-4642-b8f6-4ad456662ae5

▽ 2 : 2 elements

- key : id
- value : f942bdc6-0e75-473f-b65f-2a5a7f58b0fe

▽ 3 : 2 elements

- key : isBluetoothOn
- value : <null> { ... }

▽ 4 : 2 elements

- key : tag
- value : driver_in_my_car

▽ 5 : 2 elements

- key : startTimestamp
- value : 1619543328

▽ 6 : 2 elements

- key : vehicleId
- value : a8bf26ed-add5-4884-9bb5-2b656f7ec4f1

▽ 7 : 2 elements

- key : finishTimestamp
- value : 1619543600

▽ 1 : 2 elements

- key : "scoring"

▽ value : 6 elements

▽ 0 : 2 elements

- key : accidentness
- value : 0

▽ 1 : 2 elements

- key : version
- value : scor-1.3.0

▽ 2 : 2 elements

- key : errors
- value :

▽ 3 : 2 elements

- key : scorePercent
- value : 100

▽ 4 : 2 elements

- key : distanceMeters
- value : 1647

▽ 5 : 2 elements

- key : durationSec
- value : 228

**2. Detailed information on a trip that has been made.**

Use the method

Declaration: ScoringUserBehaviourObserver.shared.getTripScoring(for tripID: String, with completion: @escaping ([String : Any]?) -> Void)

where tripID - trip identifier.

The method returns a dictionary which contains detailed information on a specified trip:(see page 110)

```
"trip": {
      "tripData": {
        "id": UUID, // Unique identifier of the trip
```

```
        "userId": UUID, // Unique identifier of the user
        "deviceId": UUID, // Unique identifier of the device
        "vehicleId": UUID, // Unique identifier of the vehicle
        "startTimestamp": 0, // Start time of the trip (UNIX timestamp)
        "finishTimestamp": 0, // Finish time of the trip (UNIX timestamp)
        "tag": "string", // A value that the trip has been tagged with
        "isBluetoothOn": true, // Is bluetooth on
        "isBluetoothConnectionEstablished": true // Is bluetooth connection establihed
        },
      "scoring": {
        "scorePercent": 0, // A score of the driving manner safety (100% indicates safe driving)
        "durationSec": 0, // The duration of the specified period (on seconds)
        "distanceMeters": 0, // The distance covered by the vehicle during the specified period
        "accidentness": 0,// An average accidentness score during the specified period
        "version": "string", // Versions of the components that affect the score
        "errors": "string" // Error information if any scoring component fails
        }
    },
    "events": [
      {
        "timestamp": 0, // Date and time of the event (UNIX timestamp)
        "latitude": 0, // Latitude of the location point (in signed degrees format)
        "longitude": 0, // Longitude of the location point (in signed degrees format)
        "speedKph": 0, // The speed of the object at the specified time (in kilometers per hour)
        "heading": 0, // ompass direction in which the object's bow or nose is pointed (0 or 360 indicates
a direction toward true North)
        "accuracy": 0 // The accuracy of the location information
      }
    ],
    "penalties": [
      {
        "timestamp": 0, // Date and time of the event (UNIX timestamp)
        "type": "string", // The type of the event
        "durationMs": 0, // The duration (in milliseconds) of the event
        "value": 0 // Indicates the severity of the event (depends on the type)
      }
    ]
}
```

**Example:**

▽ *3 elements*

▽ *0 : 2 elements*

- key : "events"

▽ *value : 2 elements*

▽ *0 : 6 elements*

▽ *0 : 2 elements*

- key : speedKph
- value : 32.76331443786621

▽ *1 : 2 elements*

- key : timestamp

- value : 1619541688

▽ *2 : 2 elements*

- key : latitude
- value : 48.78348541259766

▽ *3 : 2 elements*

- key : longitude
- value : 44.57433319091797

▽ *4 : 2 elements*

- key : heading
- value : 42.59416961669922

▽ *5 : 2 elements*

- key : accuracy
- value : 27.90659523010254

▽ *1 : 6 elements*

▽ *0 : 2 elements*

- key : speedKph
- value : 32.76331443786621

▽ *1 : 2 elements*

- key : timestamp
- value : 1619541689

▽ *2 : 2 elements*

- key : latitude
- value : 48.78348541259766

▽ *3 : 2 elements*

- key : longitude
- value : 44.57433319091797

▽ *4 : 2 elements*

- key : heading
- value : 42.59416961669922

▽ *5 : 2 elements*

- key : accuracy
- value : 27.90659523010254

▽ *1 : 2 elements*

- key : "penalties"

▽ *value : 1 element*

▽ *0 : 4 elements*

▽ *0 : 2 elements*

- key : value
- value : -9

▽ *1 : 2 elements*

- key : timestamp
- value : 1619542035

▽ *2 : 2 elements*

- key : type
- value : Acceleration

▽ *3 : 2 elements*

- key : durationMs
- value : 1

▽ *2 : 2 elements*

- key : "trip"

▽ *value : 2 elements*

▽ *0 : 2 elements*

- key : tripData

▽ *value : 8 elements*

▽ *0 : 2 elements*

- key : userId
- value : 087fbef5-d257-434a-9a21-1220cfb797cf

▽ *1 : 2 elements*

- key : deviceId
- value : 720c62fc-73bf-4642-b8f6-4ad456662ae5

▽ *2 : 2 elements*

- key : id
- value : 76464bcc-8bf3-4e55-9d57-4eba8b3cb90d

▽ *3 : 2 elements*

- key : isBluetoothOn
- value : 1

▽ *4 : 2 elements*

- key : tag
- value : driver_in_my_car

▽ *5 : 2 elements*

- key : startTimestamp
- value : 1619541685

▽ *6 : 2 elements*

- key : vehicleId
- value : a8bf26ed-add5-4884-9bb5-2b656f7ec4f1

▽ *7 : 2 elements*

- key : finishTimestamp
- value : 1619542085

▽ *1 : 2 elements*

- key : scoring

▽ *value : 6 elements*

▽ *0 : 2 elements*

- key : accidentness
- value : 0

▽ *1 : 2 elements*

- key : version
- value : scor-1.3.0

▽ *2 : 2 elements*

- key : errors
- value :

▽ *3 : 2 elements*

- key : scorePercent
- value : 82.20716546491759

▽ *4 : 2 elements*

- key : distanceMeters
- value : 3385

▽ *5 : 2 elements*

- key : durationSec
- value : 347

**3. The weighted average value of the score and the total value of vehicle mileage over a certain period of time.**

Use the method

Declaration: ScoringUserBehaviourObserver.shared.getCommonScoring(from beginDate: Date, to endDate: Date, tag: String, with completion: @escaping ([String : Any]?) -> Void)

where

beginDate - date/time of the beginning of the period

endDate - date/time of the end of the period

tag(see page 20) - trip type

The method returns a dictionary which contains the weighted average value of the score and the total value of vehicle mileage:(see page 116)

```
{
    "scorePercent": 0, // A score of the driving manner safety (100% indicates safe driving)
    "durationSec": 0, // The duration of the specified period (on seconds)
    "distanceMeters": 0, // The distance covered by the vehicle during the specified period
    "accidentness": 0, // An average accidentness score during the specified period
    "tripsCount": 0 // A number of trips included into calculation of scoring
  },
```

**Example:**

*<Dictionary<String, Any>>*

▽ *some : 5 elements*

▽ *0 : 2 elements*

- key : "tripsCount"
- value : 10

▽ *1 : 2 elements*

- key : "accidentness"
- value : 0

▽ *2 : 2 elements*

- key : "scorePercent"
- value : 86.92312172776988

▽ *3 : 2 elements*

- key : "distanceMeters"
- value : 92094

▽ *4 : 2 elements*

- key : "durationSec"
- value : 9809

**4. Trip type change**

Changing the trip type is done by the method

Declaration: ScoringUserBehaviourObserver.shared.setTripType(for tripID: String, tag: String, with completion: @escaping (String, Bool) -> Void)

where

tripID - trip identifier

tag(see page 20) - trip type to be set.

There are four trip types:

- user as a driver of his car - “driver_in_ my_car”
- user as a driver of someone else's car - “driver_not_in_my_car”
- user as a passenger of a car- “passenger”
- user travels by public transport - “public_transport”

The method returns two variables:

- “String” - new trip type assigned to a trip;

- “Bool” - true if the trip type has been changed, false if the trip type has not been changed.

**5. Building a vehicle route/routes, getting information about dangerous road sections.**

Use the request method :

ScoringUserBehaviourObserver.shared.getRecommendedTrips(for originLat: Double, and originLon: Double, for destinationLat: Double, and destinationLon: Double, and routeTime: Int, with
completion: **@escaping** ([RecommendedTrip]?) -> Void)

where:
originLat - latitude of start trip point
originLon - longitude of start trip point
destinationLat - latitude of finish trip point
destinationLon - longitude of finish trip point
routeTime - time of trip start, UNIX timestamp

Response parameters:

Array of trips - [RecommendedTrip]

```
distance: Double = 0
duration: Int = 0
riskCountForUi: Int = 0
accidentRisk: Double = 0
lowRiskPercentage: Double = 0
normalRiskPercentage: Double = 0
highRiskPercentage: Double = 0
lowRiskDistance: Double = 0
normalRiskDistance: Double = 0
highRiskDistance: Double = 0

waypoints = [Waypoint]()

lowRoute = [String]()
normalRoute = [String]()
highRoute = [String]()

baseRoute: String = “”lowColor = UIColor()normalColor = UIColor()highColor = UIColor()baseColor =
UIColor()inactiveRouteColor = UIColor()
```

where:

lowRoute, normalRoute, highRoute - arrays of routes polyline;
baseRoute - routes polyline;

waypoints - array of points for Google navigation service:

```
"waypoints": [
        {
          "lat": ,
          "lng":
```

```
        },
            ]
```

## 4.3 Server interaction methods

When interacting with the server, a user can receive the following basic types of information:

- a list of all recorded trips made over a specified period;(see page 103)
- detailed information about a trip selected by a user;(see page 110)
- an average score value and a value of the vehicle mileage for all trips recorded over a specified period.(see page 116)
- set “tag” to the existing trip
- create or update device information
- save device log

### 4.3.1 Obtaining trips data

| | |
|---|---|
| Doc version | 1.01 - 3 November 2021 |
| Title | Scoring API |
| Short description | A list of trips for the specified user/device/vehicle during the specified period of time is output. |
| Description | The method returns a list of all recorded trips made by a user of the given vehicle (vehicleId) with the specified tag for the specified period (from - to).<br>Information on each trip includes:<br>• data on trip start and stop time;<br>• data on trip scoring statistics. |
| Contact | If you have any questions, please reach out to us info@kasko2go.com[33] |
| | |
| **Request** | |
| Request URL | https://scoring-api.kasko2go.net/api/Scoring/trips |
| Supported Request Types | application/json |
| Authorization | Use the application-provided user authentication type. |

33 mailto:info@kasko2go.com

<table>
<tr><td>Request method</td><td colspan="4">GET</td></tr>
<tr><td>Request Header</td><td colspan="4">Accept: application/json</td></tr>
<tr><td rowspan="12">Request URL Parameters</td><td colspan="4">List of parameters:</td></tr>
<tr><td>Name</td><td>Type</td><td>Required/ Optional</td><td>Description</td></tr>
<tr><td>from</td><td>date-time</td><td>Optional</td><td>Start of the period in which you want to count trips, UNIX timestamp format</td></tr>
<tr><td>to</td><td>date-time</td><td>Optional</td><td>The end of the period in which trips are to be counted, UNIX timestamp</td></tr>
<tr><td>limit</td><td>integer</td><td>Optional</td><td>No more than the specified number of records will be returned.</td></tr>
<tr><td>ofset</td><td>integer</td><td>Optional</td><td>Number of records to skip.</td></tr>
<tr><td>ascending</td><td>boolean</td><td>Optional</td><td>Specify to sort the trips in ascending order</td></tr>
<tr><td>userId</td><td>string</td><td>Required</td><td>User identifier</td></tr>
<tr><td>deviceId</td><td>string</td><td>Required</td><td>Device identifier</td></tr>
<tr><td>vehicleId</td><td>string</td><td>Required</td><td>Vehicle identifier</td></tr>
<tr><td>tag(see page 20)</td><td>string</td><td>Required</td><td>A tag describing the trip, for example, "driver_in_my_car".<br>If the tag field has the value ““, then all trips corresponding to other request parameters are returned in the response.</td></tr>
<tr><td>minTripTimeSeconds</td><td>integer</td><td>Optional</td><td>Exclude trips whose time between start and end of the trip is less than this value</td></tr>
</table>

| | | | | |
|---|---|---|---|---|
| | returnValidScoringTrip | boolean | Optional | Exclude trips with the null scoring or the scoring is zero and the number of accidents is zero |
| | **Possible combinations of the values «from» and «to»:** | | | |
| | from | to | Description | |
| | + | + | all trips in the period from … to | |
| | + | - | all trips from the beginning to the present moment are taken into account | |
| | - | + | all trips before the period specified in "to" are taken into account | |
| | - | - | all trips saved in the travel history are taken into account | |
| Example | Case 1: | | | |
| | Obtaining a list of trips and scoring data on these trips made by this user during the specified period with the specified trip characteristic:<br>userId = `40529063-56f1-47e9-b687-315263aaf8d3`<br>deviceId = `59f8154c-dcd0-4bc1-b9d7-2228480b83d2`<br>vehicleId = `f9a0d063-a9e7-432b-b414-6855622551cf`<br>tag = driver_in_my_car<br>from = 20:00 27 April 2021 --> UNIX timestamp - 1619542800<br>to = 00:00 28 April 2021 --> UNIX timestamp - 1619557200 | | | |
| | Converting the time and date from hh:mm:ss DD.MM.YYYY format to UNIX timestamp format can be done using this link https://www.freeformatter.com/epoch-timestamp-to-date-converter.html | | | |
| | **Curl**<br>curl -X GET "https://scoring-api.kasko2go.net/api/Scoring/trips?from=1619542800&to=1619557200&userId=40529063-56f1-47e9-b687-315263aaf8d3&deviceId=59f8154c-dcd0-4bc1-b9d7-2228480b83d2&vehicleId=f9a0d063-a9e7-432b-b414-6855622551cf&tag=driver_in_my_car"[34] -H "accept: application/json" | | | |

34 https://scoring-api.kasko2go.net/api/Scoring/trips?from=1619542800&to=1619557200&userId=40529063-56f1-47e9-b687-315263aaf8d3&deviceId=59f8154c-dcd0-4bc1-b9d7-2228480b83d2&vehicleId=f9a0d063-a9e7-432b-b414-6855622551cf&tag=driver_in_my_car%22

**Request URL**

https://scoring-api.kasko2go.net/api/Scoring/trips?from=1619542800&to=1619557200&userId=40529063-56f1-47e9-b687-315263aaf8d3&deviceId=59f8154c-dcd0-4bc1-b9d7-2228480b83d2&vehicleId=f9a0d063-a9e7-432b-b414-6855622551cf&tag=driver_in_my_car

**Response content:**


```
{
  "payload": {
    "limit": -1,
    "offset": 0,
    "totalCount": 2,
    "values": [
      {
        "tripData": {
          "id": "1a2da9e1-0eb0-4f31-9222-d8d8afc2feaf",
          "userId": "40529063-56f1-47e9-b687-315263aaf8d3",
          "deviceId": "59f8154c-dcd0-4bc1-b9d7-2228480b83d2",
          "vehicleId": "f9a0d063-a9e7-432b-b414-6855622551cf",
          "startTimestamp": 1619551736,
          "finishTimestamp": 1619553043,
          "tag": "driver_in_my_car",
          "isBluetoothOn": null,
          "isBluetoothConnectionEstablished": null
        },
        "scoring": {
          "scorePercent": 100,
          "durationSec": 243,
          "distanceMeters": 2039,
          "accidentness": 0,
          "version": "scor-1.3.0",
          "errors": ""
        }
      },
      {
        "tripData": {
          "id": "3bbd4ad1-4be3-41fd-9aa1-c61b7e612fa4",
          "userId": "40529063-56f1-47e9-b687-315263aaf8d3",
          "deviceId": "59f8154c-dcd0-4bc1-b9d7-2228480b83d2",
          "vehicleId": "f9a0d063-a9e7-432b-b414-6855622551cf",
          "startTimestamp": 1619543215,
          "finishTimestamp": 1619544950,
          "tag": "driver_in_my_car",
          "isBluetoothOn": null,
          "isBluetoothConnectionEstablished": null
        },
        "scoring": {
          "scorePercent": 92.66855986892294,
          "durationSec": 1348,
          "distanceMeters": 9919,
          "accidentness": 0,
          "version": "scor-1.3.0",
          "errors": ""
        }
      }
    ]
  },
```

```
  "status": "OK",
  "code": 200,
  "message": null
}
```

Response parameters

The method returns the values of the following parameters:

| Name | Type | Description |
|---|---|---|
| limit | integer | No more than the specified number of records will be returned |
| offset | integer | Number of records to skip |
| totalCount | integer | Total number of trips |
| values | Array [ | |

- tripData

| Name | Type | Description |
|---|---|---|
| id | GUID | Unique identifier of the trip. |
| userId | GUID | Unique identifier of the user. |
| deviceId | GUID | Unique identifier of the device. |
| vehicleId | GUID | Unique identifier of the vehicle. |
| startTimestamp | long | Trip start, UNIX timestamp |
| finishTimestamp | long | Trip finish, UNIX timestamp |
| tag | string | A tag describing the trip |

| | | | | |
|---|---|---|---|---|
| | | isBluetoothOn | boolean | Is bluetooth on. |
| | | isBluetoothConnectionEstablished | boolean | Is bluetooth connection establihed. |
| | | • scoring | | |
| | | Name | Type | Description |
| | | scorePercent | number | A score of the driving manner safety (100% indicates safe driving). |
| | | durationSec | integer | The duration of the specified period (on seconds). |
| | | distanceMeters | integer | The distance covered by the vehicle during the specified period, m. |
| | | accidentness | number | Accident rate index |
| | | version | string | Scoring service version |
| | | error | string | Error information if any scoring component fails. |
| | | ] | | |
| | status | string | Response status | |
| | code | intager | Error number | |
| | message | string | Response status description | |
| Execute status code | 200 – Success;<br>400 - Bad Request;<br>401 - Unauthorized;<br>429 - Too many requests. | | | |

### 4.3.2 Obtaining data on the details of the trip

| | |
|---|---|
| Doc version | 1.01 - 3 November 2021 |
| Title | Scoring API |
| Short description | Detailed information on a selected trip is output. |
| Description | The method returns detailed information on a selected trip, which includes:<br>• scoring statistics data;<br>• telemetry data collected during the trip;<br>• a list of violations and penalties during the trip. |
| Contact | If you have any questions, please reach out to us info@kasko2go.com[35] |
| | |
| **Request** | |
| Request URL | https://scoring-api.kasko2go.net/api/Scoring/[36]{id} |
| Supported Request Types | application/json |
| Authorization | Use the application-provided user authentication type. |
| Request method | GET |
| Request Header | Accept: application/json |
| Request URL Parameters | **List of parameters:** |

| Name | Type | Required/ Optional | Description |
|---|---|---|---|
| id | GUID | Required | Trip identifier |

35 mailto:info@kasko2go.com
36 https://scoring-api.kasko2go.net/api/Scoring/trips

| | | | | |
|---|---|---|---|---|
| | loadAllEvents | boolean | Optional | Identifier of trip telemetry data load.<br>Default value - false. |
| | mapToRoads | boolean | Optional | Identifier of calling the OSRM(see page 40) service for mapping telemetry data to the data of the nearest road.<br>Default value - false. |
| Example | Case 1: | | | |
| | Obtaining detailed information on the trip without telemetry data:<br>trip Id = `f4f768b6-7273-47d7-b04e-e7ecf7f26d5d` | | | |
| | **Curl**<br>curl -X GET "https://scoring-api.kasko2go.net/api/Scoring/f4f768b6-7273-47d7-b04e-e7ecf7f26d5d"[37] -H "accept: application/json" | | | |
| | **Request URL**<br>https://scoring-api.kasko2go.net/api/Scoring/f4f768b6-7273-47d7-b04e-e7ecf7f26d5d | | | |

37 https://scoring-api.kasko2go.net/api/Scoring/f4f768b6-7273-47d7-b04e-e7ecf7f26d5d%22

**Response content:**


```
{
  "payload": {
    "trip": {
      "tripData": {
        "id": "f4f768b6-7273-47d7-b04e-e7ecf7f26d5d",
        "userId": "a0970583-a9e6-4318-b47b-bddfac847bca",
        "deviceId": "a3e1e7d2-6b01-41a8-9b15-0ec9f9d1c698",
        "vehicleId": "4c17e4e0-c98d-481f-bed2-282ec24ccbfb",
        "startTimestamp": 1619244186,
        "finishTimestamp": 1619600248,
        "tag": "driver_in_my_car",
        "isBluetoothOn": null,
        "isBluetoothConnectionEstablished": null
      },
      "scoring": {
        "scorePercent": 97.99010471991784,
        "durationSec": 3108,
        "distanceMeters": 50471,
        "accidentness": 0,
        "version": "scor-1.3.0",
        "errors": ""
      }
    },
    "events": null,
    "penalties": [
      {
        "timestamp": 1619244590,
        "type": "Speeding",
        "durationMs": 1,
        "value": -4.49
      },
      {
        "timestamp": 1619247639,
        "type": "Acceleration",
        "durationMs": 1,
        "value": -4.696898
      }
    ],
    "startAddress": null,
    "endAddress": null
  },
  "status": "OK",
  "code": 200,
  "message": null
}
```


Case 2:

Obtaining detailed information on the trip and telemetry data collected during the trip:

trip Id = `f4f768b6-7273-47d7-b04e-e7ecf7f26d5d`

loadAllEvents = true

**Curl**

curl -X GET "https://scoring-api.kasko2go.net/api/Scoring/f4f768b6-7273-47d7-b04e-e7ecf7f26d5d?loadAllEvents=true"[38] -H "accept: application/json"

**Request URL**

https://scoring-api.kasko2go.net/api/Scoring/f4f768b6-7273-47d7-b04e-e7ecf7f26d5d?loadAllEvents=true

**Response content:**

response_1620119479620.j…

(see page 110)

Response parameters

The method returns the values of the following parameters:

| Name | Type | Description |
|---|---|---|
| trip | Array [ | |
| | • tripData | |

| Name | Type | Description |
|---|---|---|

38 https://scoring-api.kasko2go.net/api/Scoring/f4f768b6-7273-47d7-b04e-e7ecf7f26d5d?loadAllEvents=true%22

| | | |
|---|---|---|
| id | GUID | Unique identifier of the trip. |
| userId | GUID | Unique identifier of the user. |
| deviceId | GUID | Unique identifier of the device. |
| vehicleId | GUID | Unique identifier of the vehicle. |
| startTimestamp | long | Trip start time, UNIX timestamp |
| finishTimestamp | long | Trip finish time, UNIX timestamp |
| tag(see page 20) | string | A tag describing the trip |
| isBluetoothOn | boolean | Is bluetooth on. |
| isBluetoothConnectionEstablished | boolean | Is bluetooth connection establihed. |

- scoring

| Name | Type | Description |
|---|---|---|
| scorePercent | number | A score of the driving manner safety (100% indicates safe driving). |
| durationSec | integer | The duration of the specified period (in seconds). |
| distanceMeters | integer | The distance covered by the vehicle during the specified period, m. |
| accidentness | number | Accident rate index |
| version | string | Scoring service version |
| error | string | Error information if any scoring component fails. |

]

| | |
|---|---|
| events | Array [ |
| | <table><tr><th>Name</th><th>Type</th><th>Description</th></tr><tr><td>timestamp</td><td>integer</td><td>Event time, UNIX timestamp</td></tr><tr><td>latitude</td><td>number</td><td>Latitude of the location point (in signed degrees format).</td></tr><tr><td>longitude</td><td>number</td><td>Longitude of the location point (in signed degrees format).</td></tr><tr><td>speedKph</td><td>number</td><td>The speed of the object at the specified time (in kilometers per hour).</td></tr><tr><td>heading</td><td>number</td><td>Compass direction in which the object's bow or nose is pointed (0 or 360 indicates a direction toward true North).</td></tr><tr><td>accuracy</td><td>number</td><td>Accuracy in metres. Maximum deviation (margin of error) of GPS coordinates from the true value.</td></tr></table> |
| | ] |
| penalties | Array [ |
| | <table><tr><th>Name</th><th>Type</th><th>Description</th></tr><tr><td>timestamp</td><td>integer</td><td>Event time, UNIX timestamp</td></tr><tr><td>type</td><td>string</td><td>Violation type (acceleration, speeding, etc.)</td></tr><tr><td>durationMs</td><td>integer</td><td>The duration (in milliseconds) of the event.</td></tr><tr><td>value</td><td>number</td><td>Number of penalties</td></tr></table> |
| | ] |

| | | | | |
|---|---|---|---|---|
| | | startAddress | string | Localized start address |
| | | endAddress | string | Localized end address |
| | status | string | Response status | |
| | code | intager | Error number | |
| | message | string | Response status description | |
| Execute status code | 200 – Success;<br>400 - Bad Request;<br>401 - Unauthorized;<br>429 - Too many requests. | | | |

### 4.3.3 Obtaining the average score and recorded mileage

| | |
|---|---|
| Doc version | 1.01 - 3 November 2021 |
| Title | Scoring API |
| Short description | The weighted average score and mileage for all trips of a selected vehicle with the selected trip characteristic for the specified period of time is output. |
| Description | The method returns:<br>• the weighted average score for all trips of the given vehicle (vehicleId) with the specified tag for the specified period (from - to). Scoring is weighted by trip distance (not weighted average).<br>• the mileage for all trips of the given vehicle (vehicleId) with the specified tag for the specified period (from - to).<br>Example: the car made two trips:<br>1. distance - 5000 m, score - 100%;<br>2. distance - 4000 m, score - 90%).<br>The weighted average score for the two specified trips of this car will be calculated as follows,<br>distanseSum = 5000 + 4000 = 9000 m<br>scoring = (5000 * 100 + 4000 * 90) / distanseSum = 95,56% |
| Contact | If you have any questions, please reach out to us info@kasko2go.com[39] |

39 mailto:info@kasko2go.com

| | |
|---|---|
| **Request** | |
| Request URL | https://scoring-api.kasko2go.net/api/Scoring |
| Supported Request Types | application/json |
| Authorization | Use the application-provided user authentication type. |
| Request method | GET |
| Request Header | Accept: application/json |
| Request URL Parameters | **List of parameters:** |

| Name | Type | Required/Optional | Description |
|---|---|---|---|
| from | date-time | Optional | Start of the period in which you want to count trips, UNIX timestamp |
| to | date-time | Optional | End of the period in which trips are to be counted, UNIX timestamp |
| userId | string | Required | Unique identifier of the user |
| deviceId | string | Required | Unique identifier of the device |
| vehicleId | string | Required | Unique identifier of the vehicle |
| tag | string | Required | A tag describing the trip, for example, "driver_in_my_car".<br>If the tag field has the value "", then trips corresponding to other request parameters are returned in the response. |
| minTripTimeSeconds | integer | Optional | Exclude trips whose time between start and end of the trip is less than this value |

| | | | |
|---|---|---|---|
| returnValidScoringTrip | boolean | Optional | Exclude trips with the null scoring or the scoring is zero and the number of accidents is zero |

**Possible combinations of the values «from» and «to»:**

| from | to | Description |
|---|---|---|
| + | + | all trips in the period from … to |
| + | - | all trips from the beginning to the present moment are taken into account |
| - | + | all trips before the period specified in "to" are taken into account |
| - | - | all trips saved in the travel history are taken into account |

Example

<u>Case 1:</u>

Calculate the weighted average value of the score and mileage for all trips of this car during the specified period with the specified trip characteristic:

userId = `40529063-56f1-47e9-b687-315263aaf8d3`

deviceId = `59f8154c-dcd0-4bc1-b9d7-2228480b83d2`

vehicleId = `f9a0d063-a9e7-432b-b414-6855622551cf`
tag = driver_in_my_car
from = 20:00 27 April 2021 --> UNIX timestamp - 1619542800

to = 00:00 28 April 2021 --> UNIX timestamp - 1619557200

Converting the time and date from hh:mm:ss DD.MM.YYYY format to UNIX timestamp format can be done using this link https://www.freeformatter.com/epoch-timestamp-to-date-converter.html

**Curl**

curl -X GET "https://scoring-api.kasko2go.net/api/Scoring?from=1619542800&to=1619557200&userId=40529063-56f1-47e9-b687-315263aaf8d3&deviceId=59f8154c-dcd0-4bc1-b9d7-2228480b83d2&vehicleId=f9a0d063-a9e7-432b-b414-6855622551cf&tag=driver_in_my_car"[40] -H "accept: application/json"

**Request URL**

https://scoring-api.kasko2go.net/api/Scoring?from=1619542800&to=1619557200&userId=40529063-56f1-47e9-b687-315263aaf8d3&deviceId=59f8154c-dcd0-4bc1-b9d7-2228480b83d2&vehicleId=f9a0d063-a9e7-432b-b414-6855622551cf&tag=driver_in_my_car

**Response content:**


```
{
  "payload": {
    "scorePercent": 93.9186691202414,
    "durationSec": 1591,
    "distanceMeters": 11958,
    "accidentness": 0,
    "tripsCount": 2
  },
  "status": "OK",
  "code": 200,
  "message": null
}
```


Case 2:

Calculate the weighted average value of the score and mileage for all trips of this driver starting from the specified time:

userId = `40529063-56f1-47e9-b687-315263aaf8d3`

deviceId = `59f8154c-dcd0-4bc1-b9d7-2228480b83d2`

vehicleId = `f9a0d063-a9e7-432b-b414-6855622551cf`
tag = driver_in_my_car

from = 00:00 15 April 2021 --> UNIX timestamp - 1618434000

40 https://scoring-api.kasko2go.net/api/Scoring?from=1619542800&to=1619557200&userId=40529063-56f1-47e9-b687-315263aaf8d3&deviceId=59f8154c-dcd0-4bc1-b9d7-2228480b83d2&vehicleId=f9a0d063-a9e7-432b-b414-6855622551cf&tag=driver_in_my_car%22

<table>
<tr><td></td><td>

**Curl**

curl -X GET "https://scoring-api.kasko2go.net/api/Scoring?from=1618434000&userId=40529063-56f1-47e9-b687-315263aaf8d3&deviceId=59f8154c-dcd0-4bc1-b9d7-2228480b83d2&vehicleId=f9a0d063-a9e7-432b-b414-6855622551cf&tag=driver_in_my_car"[41] -H "accept: application/json"

</td></tr>
<tr><td></td><td>

**Request URL**

https://scoring-api.kasko2go.net/api/Scoring?from=1618434000&userId=40529063-56f1-47e9-b687-315263aaf8d3&deviceId=59f8154c-dcd0-4bc1-b9d7-2228480b83d2&vehicleId=f9a0d063-a9e7-432b-b414-6855622551cf&tag=driver_in_my_car

</td></tr>
<tr><td></td><td>

**Response content:**


```
{
  "payload": {
    "scorePercent": 81.09621404713131,
    "durationSec": 70330,
    "distanceMeters": 625701,
    "accidentness": 0,
    "tripsCount": 66
  },
  "status": "OK",
  "code": 200,
  "message": null
}
```


</td></tr>
<tr><td>Response parameters</td><td>

The method returns the values of the following parameters:

| Name | Type | Description |
|---|---|---|
| scorePercent | number | The score of the driving manner safety (100% indicates safe driving). |
| durationSec | integer | The duration of the specified period (on seconds). |
| distanceMeters | integer | The distance covered by the vehicle during the specified period, m. |

</td></tr>
</table>

41 https://scoring-api.kasko2go.net/api/Scoring?from=1618434000&userId=40529063-56f1-47e9-b687-315263aaf8d3&deviceId=59f8154c-dcd0-4bc1-b9d7-2228480b83d2&vehicleId=f9a0d063-a9e7-432b-b414-6855622551cf&tag=driver_in_my_car%22

| | | | |
|---|---|---|---|
| | accidentness | number | Accident rate index |
| | tripsCount | integer | A number of trips included into calculation of the score |
| | status | string | Response status |
| | code | integer | Error number |
| | message | string | Response status description |
| Execute status code | 200 – Success;<br>400 - Bad Request;<br>401 - Unauthorized;<br>429 - Too many requests. | | |

## 4.4 SDK performance methods

Open source solution SDK is designed:

- for acquaintance with the proposed Open source solution product;
- for acquaintance with the technique for determining vehicle driving start and finish(see page 121);

Open source solution SDK includes:

- SDK for smartphones with Android operating system;(see page 48)
- SDK for smartphones with iOS operating system;(see page 67)
- Binary data transfer protocol(see page 29).

### 4.4.1 Start and stop of trip detection

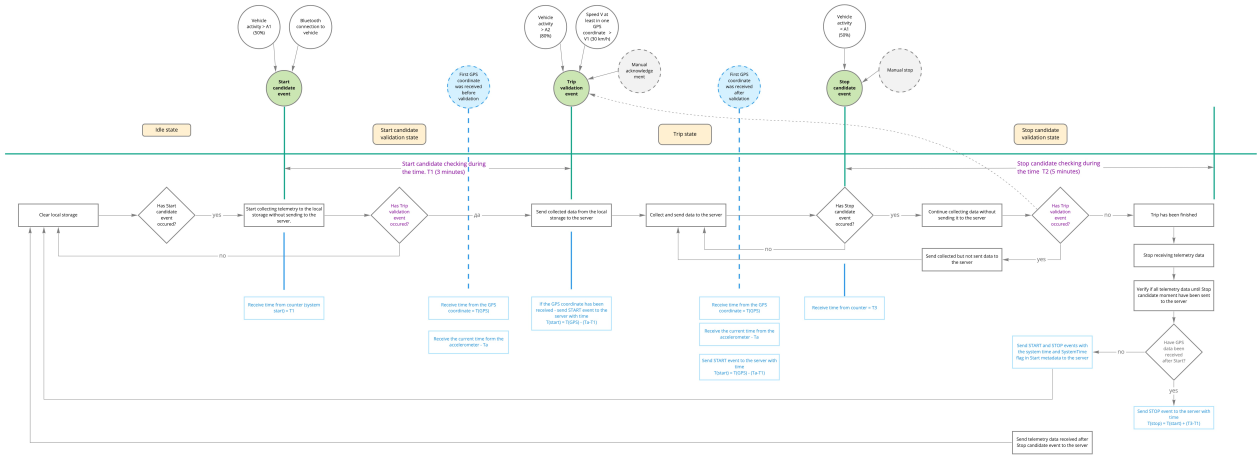
One of the important elements of the Open source solution is a system of automatic trip detection. Trip detection in done by a user smartphone with a built-in accelerometer and a GPS satellite system module. A user should install a mobile application with the GPS tracker function on the smartphone.

The mobile application automatically detects the vehicle driving start and finish moments. A "trip" is formed within the time interval from the driving start and driving finish; the mobile application collects and transmits telemetry data to the processing server for further score calculation.

When considering the procedure for detecting the moments of vehicle driving start and finish, it should be taken into account that the GPS system and the smartphone accelerometer work with different time scales:

- GPS system time is connected to the Coordinated Universal Time (UTC);
- system time of accelerometer is the time which starts from the moment of the mobile device operating system start.

A flowchart of the algorithm for the system of automatic detecting the vehicle driving start and finish, as well as formation of trips, is given in the Figure.



When automatically determining the beginning and end of a trip, we should apply the principle of impossible missing of a trip, even if it has been mistakenly identified (formed).

The initial operational state of the system is the idle mode (idle). In this condition the mobile application analyses the following indicators (triggers) of the driving start (Start candidate) with the periodicity $T_0$ :

- smartphone motion detector according to accelerometer data at the level $A_1$ and higher;
- connecting a smartphone to the Bluetooth receiver of the vehicle.

If at least one of the driving start triggers (Start candidate) is enabled, the system goes to the state of validating the driving start (Start candidate validation), in which:

- the current value of the mobile device system time is recorded;
- the current value of accelerometer system time is recorded (sensorEvent.timestamp is a timestamp related to the system startup time);
- the countdown timer $T_1$, which determines the maximum duration period of the driving start validation state, is started;
- the work with the GSM module of the smartphone is initialised;
- collection and accumulation of telemetry data in the local database of the mobile application begins without sending it to the server.

During the period $T_1$ , indicators that confirm the driving (Trip validation) are being analysed

- smartphone motion detector according to accelerometer data at the level $A_2$ and higher;
- driving speed in at least one point in space has exceeded the value $V_1$ according to the GPS system data.

If during the period $T_1$ none of the triggers Trip validation has been enabled, then the accumulated telemetry data is deleted and the system returns to its initial idle state.

If during the period $T_1$ at least one of the triggers Trip validation is enabled, then:

- the system goes to the Trip state;
- information on the Start event is sent to the server;
- telemetry data accumulated in the local database of the mobile application for the period $RT_1$ is sent to the server;
- transmission of the current telemetry data to the server begins;
- monitoring the occurance of the driving stop event begins (Stop candidate) - smartphone motion detector according to accelerometer data at the level $A_1$ and lower.

If at the moment of Start candidate validation (system entering the state Trip) the smartphone has established a connection and begun to receive data from the GPS satellite system, then transferring information about the Start event to the server is accompanied by the transmission of the Start event occurence time, which is determined by the formula:

$$T_{Start} = T_{GPS} - (T_a - T_1),$$

where $T_{GPS}$ - the time of receiving the first point of coordinates of the smartphone location from the GPS system;

$T_a$ - the value of the smartphone accelerometer system time at the moment of receiving the first point of smartphone location coordinates from the GPS system;

$T_1$ - the value of the smartphone accelerometer system time at the moment of the Start candidate event occurance.

If at the moment of Start candidate validation (system entering the state Trip) the smartphone has no connection with the GPS satellite system, then the event Start is sent to the server with the value of the accelerometer system time $T_1$ and the flag SystemTime is set in the Start metadata. Then the system continues to transfer available current telemetry data to the server and continues to monitor the establishment of a connection with GPS satellites.

If while driving, in the Trip state, the trigger Stop candidate is enabled, then the system goes to the state of driving stop validation (Stop candidate validation) and starts validating the fact of trip identification. Being in the Stop candidate validation state, the system performs the following actions:

- it records the current value of the smartphone accelerometer system time $T_3$;
- the countdown timer $T_2$, which determines the duration of the period of driving stop validation state, is started ;
- the telemetry data (if there is any) received up to the moment of Stop candidate trigger enabling (moment of time $T_3$) is sent to the server.
- once the Stop candidate trigger has been enabled, telemetry data continues to be collected, however, it is not sent to the server but accumulated in the local database of the mobile application;
- during the period $T_2$ the indicators confirming driving (Trip validation) are being analysed.

If during the period $T_2$ at least one of the triggers Trip validation enables, then it is considered that the trip is continued, in addition:

- the telemetry data accumulated in the local database after the Stop candidate event occurance is sent to the server;
- the system returns to the Trip state;
- the value of the countdown timer $T_2$ resets.

If during the period $T_2$ none of the Trip validation triggers enables, then it is considered that the trip has been finished, in addition:

- a message on the Stop event occrance is sent to the server;
- sending the message about the Stop event to the server is accompanied by sending the time of the Stop event occurance, which is determined by the formula:

$$T_{Stop} = T_{Start} + (T_3 - T_1),$$

where $T_{Start}$ - time of the Start event occurance;

$T_3$ - the value of the smartphone accelerometer system time at the moment of the Stop candidate event occurance;

$T_1$ - the value of the smartphone accelerometer system time at the moment of the Start candidate event occurance.

- the accumulation of telemetry data in the local database stops;
- telemetry data (if there is any) accumulated before the Stop candidate event occurance is sent to the server from the local database;
- the system returns to its initial idle state.

If at the moment of the Stop candidate validation the smartphone has no connection with the GPS satellite system, then the Stop event is sent to the server with the value of accelerometer system time $T_3$ and the flag SystemTime is set in the Stop metadata.

Values of operating parameters and coefficients

| Parameter | Description | Measurement units | Default value |
|---|---|---|---|
| $T_0$ | Periodicity of obtaining a motion detector indicator from the smartphone motion detection service | sec | 10 |
| $T_1$ | Maximum duration of trip validation state (Start candidate validation) | sec | 180 |
| $T_2$ | Duration of trip validation state (Stop candidate validation) | sec | 300 |
| $A_1$ | The threshold for motion probability “In Vehicle“ for Activity_Auto_candidate in: | | |
| | • OS Android (getConfidence[42]) | % | 50 |
| | • OS iOS (CMMotionActivityConfidence[43]) | - | medium |
| $A_2$ | The threshold for motion probability “In Vehicle“ in Android for Activity_Auto_validated in: | | |

42 https://developers.google.com/android/reference/com/google/android/gms/location/DetectedActivity#IN_VEHICLE
43 https://developer.apple.com/documentation/coremotion/cmmotionactivityconfidence

| Parameter | Description | Measurement units | Default value |
|---|---|---|---|
| | • OS Android (getConfidence[44]) | % | 80 |
| | • OS iOS (CMMotionActivityConfidence[45]) | - | high |
| $V_1$ | The threshold driving speed "In Vehicle" to confirm the trip by GPS system data | km/h | 30 |

Possible system states and transitions between states

| States | Idle | Start candidate validation | Trip | Stop candidate validation |
|---|---|---|---|---|
| **Idle** | - | by Start candidate event | - | - |
| **Start candidate validation** | during $RT_1$ the Trip validation event HAS NOT occured - the trip has not started | - | during $RT_1$ the Trip validation event has occured - the trip has been started. | - |
| **Trip** | - | - | - | by Stop candidate event |
| **Stop candidate validation** | during $RT_2$ the Trip validation event HAS NOT occured - the trip is finished. | - | during $RT_2$ the Trip validation event has occured - the trip is being continued. | - |

### 4.4.2 Penalties and scores

The Open source solution provides for the accrual of penalties(see page 126) for a vehicle driver for 'unsafe' driving. Actions that pose a potential hazard on the road, for which a driver is accued penalties, include:

- Speeding
- Rapid acceleration
- Continuous long trips

44 https://developers.google.com/android/reference/com/google/android/gms/location/DetectedActivity#IN_VEHICLE
45 https://developer.apple.com/documentation/coremotion/cmmotionactivityconfidence

The Open source solution calculates two types of scores:

- Score of an individual trip
- Weighted average score of several trips

The score for one trip is calculated using the following formula:

$$Scoring = \frac{\max Score - Penalties}{\max Score}$$

| | | |
|---|---|---|
| where | $\max Score$ | the maximum possible score for one trip<br>(calculated in proportion to the trip length) |
| | $Penalties$ | the sum of all kinds of penalties(see page 126) for one trip |

The weighted average score for several trips is calculated as follows:

$$Scoring_{WA} = \frac{\sum_{1}^{n} \max Score - \sum_{1}^{n} Penalties}{\sum_{1}^{n} \max Score}$$

| | | |
|---|---|---|
| where | $\sum_{1}^{n} \max Score$ | the maximum possible score for several trips<br>(calculated in proportion to the length of trips ) |
| | $\sum_{1}^{n} Penalties$ | the sum of all kinds of penalties(see page 126) for several trips |
| | *n* | the number of trips. |

The penalties accumulated by a driver for various types of 'unsafe' driving and calculated values of the trips score are transferred to the Scoring (SQL)(see page 36) database for storing.

### 4.4.2.1 Penalty types

**Penalties for excessive vehicle speeding:**

- Vehicle speed is measured in km/h.
- Penalties are accrued only if the vehicle speed exceeds maximum allowed speed at a road section by more than 20 km/h.
- Speeding on the road sections with the same speed limit is accrued no more than 1 time per minute.
- To calculate a trip score, a pentalty with the highest absolute value is selected out of individual penalties collected during a trip.

The formula for calculating a penalty for vehicle speeding takes into account the following factors:

- Accident rate of a road section
- Value of exceeding the permitted speed limit on a road section over allowed +20 km/h

- Time of day
- Season
- Speeding violations near a Safety Object(see page 20)

Examples of Safety Objects.

| ID | Type | Description | Distance (m) |
|---|---|---|---|
| 1 | SpeedControlCamera | Speed Control Camera | 100 |
| 2 | AccidentProneIntersection (dangerous crossroad) | Accident-prone area | 100 |
| 3 | RailCrossing | Railway crossing | 100 |
| 4 | SchoolZone | School zone | 100 |
| 5 | LyingPoliceman (speed bump) | Speed bump | 50 |
| 7 | DangerousCorner (Dangerous turn) | Dangerous corner | 100 |
| 8 | DangerousRoadArea (Dangerous track section) | Dangerous road area | 100 |
| 9 | PedestrianCrosswalk | Pedestrian Crosswalk | 50 |

**Penalties for rapid acceleration:**

- are calculated in the case when the smartphone accelerometer recorded an excess of a set acceleration rate threshold (acceleration/deceleration);
- currently provide four threshold values of acceleration;
- are calculated taking into account the current vehicle speed at the moment of acceleration detection.

**Penalties for long-duration trips** have a fixed value (-10 points) and are accrued in the case when trip duration exceeds 2 hours.

### 4.4.2.2 Mobile penalty calculation

Transmission of accelerometer data and GPS data should start when a trip start is detected and stop transmission after a trip stop is detected.

Implement next classes and functions:

**Function getSensor**

Function getSensor: get thold sensor structure from specific sensor accelerometer data for android and OS platform.

sensor = getSensor(sensor.x, sensor.y, sensor.z, sensor.timestamp, android)

- TimeUsecConst = TimeUTC - sensor.timestamp // TimeUsecConst calculated once at time, when the start of the trip is detected in nanoseconds
- TimeUsec' = TimeUsecConst + sensor.timestamp

`getSensor` input data:

| Parametr | Type | Purpose |
|---|---|---|
| sensor.x | float32 | x-axis acceleration |
| sensor.y | float32 | y-axis acceleration |
| sensor.z | float32 | z-axis acceleration |
| TimeUsec' | int64 | calculated accelerometer timestamp (TimeUsecConst + sensor.timestamp) |
| android | bool | true if OS platform is android, false - ios |

**Function getThresholdInterface**

Function getThresholdInterface: get thold main class.

thold main class has next methods:

- settingsFromJSON(config) - set users config for calculation penalty

`settingsFromJSON` input data:

| Parametr | Type | Purpose |
|---|---|---|
| config | string | example: `"{\"InitThreshold\":[{\"Detect\":{\"D\":1.019367991845056,\"T\":300000,\"RID\":0},\"InfelicityBegin\":0,\"InfelicityAfter\":300000,\"THBottom\":1.019367991845056},{\"Detect\":{\"D\":1.0642201834862384,\"T\":500000,\"RID\":0},\"InfelicityBegin\":0,\"InfelicityAfter\":300000,\"THBottom\":1.0642201834862384},{\"Detect\":{\"D\":1.1141692150866462,\"T\":300000,\"RID\":0},\"InfelicityBegin\":0,\"InfelicityAfter\":300000,\"THBottom\":1.1141692150866462},{\"Detect\":{\"D\":1.165137614678899,\"T\":300000},\"InfelicityBegin\":0,\"InfelicityAfter\":300000,\"THBottom\":1.165137614678899}],\"AccelerationSpeedMultiplier\":{\"Distances\":[0,4.9,5,31,32767],\"Values\":[0,0,1,3,3],\"TimeBetweenPenaltys\":8,\"TypeMultiplier\":1,\"SleepTHNumber\":3,\"SleepTHDuration\":5,\"TimeFilterInputSensor\":100,\"RemovedPenaltyNumber\":[1],\"RemovedPenaltyMinDuration\":[2]}}"`<br>`InitThreshold` - config for thresholds (0,1,2,3 consistently)<br>`AccelerationSpeedMultiplier` - config for acceleration speed multiplier |

- calculate(sensor, positiveOnly) - calculation of penalty, returning 2 type of penalty: unfiltered and filtered.

`calculate` input data:

| Parametr | Type | Purpose |
|---|---|---|
| sensor | thold.Sensor | getSensor output type from specific sensor accelerometer data and OS platform. |
| positiveOnly | bolean | true - take negative values (dips / pits) when calculating speed and distance<br>false - disregard negative values (dips / pits) when calculating speed and distance |

`calculate` output data:

| Parametr | Type | Purpose |
|---|---|---|
| response | *thold.Resp | response for penalty structure |

`thold.Resp` structure:

| Parametr | Type | Purpose |
|---|---|---|
| **Parametr** | **Type** | **Purpose** |
| thPenalty | *thold.THPenaltyItem | filtered penalty Item (config implement filter parameters) |
| thPenaltyUnfiltred | *thold.THPenaltyItem | unfiltered penalty Item |

`thold.THPenaltyItem` structure:

| Parametr | Type | Purpose |
|---|---|---|
| number | int | number of threshold penalty |
| d | float32 | duration in second from previous penalty |
| distance | float32 | distance in metres for penalty |
| durations | float32 | durations in seconds for penalty |
| startU | int64 | timestamp in microseconds of sensor for penalty |
| value | float32 | value of penalty |

#### 4.4.2.3 Server penalty calculation

Some of trip penalties are calculated on the server side, and some on the smartphone side.

A smartphone sends GPS data, accelerometer data and trip penalties calculated by the smartphone to the server.

**GPS structure data:**

| Parametr | Type | Purpose |
|---|---|---|
| TimeUTC | int32 | GPS timestamp in seconds UTC |
| Latitude | float32 | GPS latitude |
| Longitude | float32 | GPS longitude |
| SpeedMps | float32 | GPS speed in mps |

| Parametr | Type | Purpose |
|---|---|---|
| Heading | float32 | GPS heading |
| Accuracy | float32 | GPS accuracy |

**Accelerometer sensor structure data:**

| Parametr | Type | Purpose |
|---|---|---|
| TimeUsec | int64 | Accelerometer timestamp in microseconds |
| x | float32 | x-axis acceleration (for IOS: x*9.81) |
| y | float32 | y-axis acceleration (for IOS: y*9.81) |
| z | float32 | z-axis acceleration (for IOS: z*9.81) |

For all values, you can change the data type if the specified precision is preserved.

TimeUsec is needed to bind GPS data with accelerometer data. When the start of the trip is detected, you need to calculate once constant TimeUsecConst = TimeUTC - sensor.timestamp in nano seconds.

Requirements:

- GPS coordinates must be taken every 200 meters
- The last known timestamp of accelerometer is saved into GPS TimeUsec value in microseconds with transform : TimeUsec = (TimeUsecConst + sensor.timestamp)/1000
- Accelerometer values are taken at intervals SENSOR_DELAY_UI[46] (60 mc)
- Accelerometer data must be clearly defined for which device it was received by Android or iOS.

**Accelerometer penalty structure data:**

| Parametr | Type | Purpose |
|---|---|---|
| number | byte | number of threshold penalty (value from 1 to 3) |
| d | float32 | duration in second from previous penalty |
| distance | float32 | distance in metres for penalty |
| durations | float32 | durations in seconds for penalty |

46 https://developer.android.com/reference/android/hardware/SensorManager#SENSOR_DELAY_UI

| Parametr | Type | Purpose |
|---|---|---|
| startU | int64 | timestamp (TimeUsec) in microseconds of candidate sensor for penalty |
| value | float32 | value of penalty |

**The protocol data structure:**

The protocol data structure based on Client-server data exchange(see page 29) with a new types.

Since the accelerometer data is taken more often than GPS coordinates, it must be sent as a separate message. The data structure of this message:

| Field | Size in bytes | Size full |
|---|---|---|
| TimeUsec | 8 | 20 bytes |
| x | 4 | |
| y | 4 | |
| z | 4 | |
| TimeUsec | 8 | 20 bytes |
| x | 4 | |
| y | 4 | |
| z | 4 | |
| … repeats until GPS coordinates appear | | |

When the GPS coordinates appear, you can send them in a separate message:

| Field | Size in bytes | Size full |
|---|---|---|
| TimeUTC | 4 | 24bytes |
| Latitude | 4 | |
| Longitude | 4 | |

| Field | Size in bytes | Size full |
|---|---|---|
| SpeedMps | 4 | |
| Heading | 4 | |
| Accuracy | 4 | |

When a penalty appears, you need to send it in a separate message:

| Field | Size in bytes | Size full |
|---|---|---|
| number | 1 | 25 bytes |
| d | 4 | |
| distance | 4 | |
| durations | 4 | |
| startU | 8 | |
| value | 4 | |

### 4.4.3 UI localization

By default, the language of the Open source solution application interface is English.

The application has an option for changing user interface language - localisation of application.

Mock-ups of images displayed on a smartphone contain fields for displaying various information messages. A list of variants of information messages displayed on the smartphone screen is given in the mobile application as a file with pairs of values “key - value”:

- key - notation key of a text field on the mobile application interface mock-up; the value of this filed is displayed on the smartphone screen during mobile application operation; it does not depend on the choice of user interface language;
- value - text value corresponding to the key for a selected language of the mobile application interface.

Key Value

"autostart_label_forgetdevice" = "Forget this device";
"mainpopup_header_bluetooth_off" = "Turn on Bluetooth";
"label_system_fitness_access" = "We only use the motion sensor to rate your driving.";

A file format of the value pairs “key - value” used in the mobile application depends on the operation system of a smartphone:

- for Android - in .xml format;
- for iOS - in .strings format.

Localization of a mobile application can be performed in two ways:

- manually by creating a file of value pairs “key - value” for a certain language of the user interface;
- automatically by organising interaction with the on-line translation service.

In this project the localisation of mobile application is performed by organising an interaction with the Localize[47] external service — a service for translating web applications and sites in automatic mode.

47 https://lokalise.com