



## Adapting driver behaviour for lower emissions

# MODALES D5.3: Low-emission driving assistance tools (support document)

<b>WORK PACKAGE 5</b>	Guidelines and tools for low-emission training
<b>TASK 5.3</b>	Low emission driving assistance tools
<b>AUTHORS</b>	Sébastien Faye (Editor and Main author), Ramiro Camino, Joan Baixauli, Maël Cornil – LIST ( <i>LUXEMBOURG INSTITUTE OF SCIENCE AND TECHNOLOGY</i> ) Engin Özatay, Orhan Alankuş – OKAN ( <i>ISTANBUL OKAN UNIVERSITY</i> ) Nikos Dimokas, Dimitri Margaritis – CERTH ( <i>HELLENIC INSTITUTE OF TRANSPORT</i> )
<b>DISSEMINATION LEVEL</b>	Public (PU)
<b>STATUS</b>	Final, approved by the European Commission
<b>DUE DATE</b>	31/08/2022
<b>DOCUMENT DATE</b>	05/12/2022
<b>VERSION NUMBER</b>	1.0



This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 815189.

## Quality control

	Name	Organisation	Date
<b>Editor:</b>	Sébastien Faye	LIST	02/12/2022
<b>Peer review 1:</b>	Guillaume Saint Pierre	CEREMA	10/11/2022
<b>Peer review 2:</b>	Ying Li	DYNNOTEQ	12/11/2022
<b>Authorised by (Technical Coordinator):</b>	Dimitris Margaritis	CERTH	02/12/2022
<b>Authorised by (Project Coordinator):</b>	Andrew Winder	ERTICO	05/12/2022
<b>Submitted by:</b>	Andrew Winder	ERTICO	05/12/2022

## Legal disclaimer

This document is issued within the framework of and for the purpose of the MODALES project. This project has received funding from the European Union's Horizon 2020 Framework Programme, through the European Climate, Infrastructure and Environment Executive Agency (CINEA) under the powers delegated by the European Commission and under Grant Agreement No. 815189. Opinions expressed and arguments employed herein do not necessarily reflect the official views of the European Commission. Neither the European Commission nor the MODALES partners bear any responsibility for any use that may be made of the information contained herein. This document and its content are the property of the MODALES Consortium. All rights relevant to this document are determined by the applicable laws. Access to this document does not grant any right or license on the document or its contents. MODALES partners may use this document in conformity with the MODALES Consortium Grant Agreement provisions.

## Table of contents

Quality control .....	2
Legal disclaimer .....	3
Table of contents .....	4
Index of Figures .....	6
Index of Tables .....	6
List of abbreviations and acronyms .....	7
Executive Summary .....	8
1. Introduction .....	9
1.1. Background .....	9
1.2. Purpose and scope .....	9
1.3. MODALES WP5 on low-emission driving tools .....	10
1.4. Scope and intended audience of this deliverable .....	11
1.5. Deviations from the Description of Action .....	11
2. Overview of MODALES tools .....	12
2.1. The mobile assistant .....	12
2.2. The project's dashboard .....	13
2.3. The internal reporting platform for trial site and data management .....	14
2.4. Overview of development responsibilities .....	15
3. Technical implementation .....	16
3.1. Specifications .....	16
3.2. High-level architecture .....	16
3.3. OBD data collection .....	16
3.4. Client Side .....	17
3.5. Server Side .....	20
3.6. External Services .....	21
3.7. Deployment .....	22
3.8. Databases .....	23
4. Focus on: the active recommendation system .....	26
4.1. Instantaneous user acceleration .....	26
4.2. Real-time recommendations .....	27
5. The passive recommendation system .....	29
5.1. Scoring system .....	29

5.2. Recommendation system .....	33
6. The web dashboard .....	35
7. Instructions for end-users: how to use the mobile app? .....	43
8. Privacy and GDPR management .....	45
8.1. Identity of the joint controllers.....	45
8.2. Categories of personal data we collect and purposes of processing .....	45
9. Conclusions .....	47

## Index of Figures

Figure 1: Screensaps of the app's data collection module .....	13
Figure 2: Dashboard web application .....	13
Figure 3: Reporting platform: dataset download for each trial site .....	14
Figure 4: Reporting platform: OBD data collection analysis .....	14
Figure 5: High-level architecture of the mobile app .....	16
Figure 6: Illustration of the link between the car and the mobile app .....	17
Figure 7: Database structure of the MODALES server .....	23
Figure 8: Smart axes (source: medium.com) .....	26
Figure 9: Active recommendation system .....	28
Figure 10: Scoring methodology for passive recommendations .....	29
Figure 11: Threshold values determined for passenger cars .....	32
Figure 12: example for an acceleration profile .....	32
Figure 13: Penalty factors .....	32
Figure 14: Passive recommendation system .....	34
Figure 15: Dashboard's login page .....	35
Figure 16: Overall dashboard .....	36
Figure 17: Dashboard per Region .....	36
Figure 18: Warnings .....	37
Figure 19: User's dashboard .....	37
Figure 20: Entity Relational Model of dashboard's data storage .....	38

## Index of Tables

Table 1: Overview of the software components developed in the project and leadership.....	15
Table 2: Speed Range vs Acceleration and Deceleration Thresholds .....	27
Table 3: Recommendation system .....	33
Table 4: Dashboard API for receiving data from the server .....	39

## List of abbreviations and acronyms

Abbreviation	Meaning
<b>AI</b>	Artificial Intelligence
<b>API</b>	Application Programming Interface
<b>CINEA</b>	European Climate, Infrastructure and Environment Executive Agency
<b>CMT</b>	Core Management Team
<b>CSS3</b>	Cascading Style Sheets
<b>DALED</b>	Driving Assistance app for Low-Emission Driving
<b>DoA</b>	Description of Action (MODALES project contract with EC/CINEA)
<b>EC</b>	European Commission
<b>EOBD</b>	European On-Board Diagnostics
<b>EU</b>	European Union
<b>FTP</b>	File Transfer Protocol
<b>GPS</b>	Global Positioning System
<b>HTML5</b>	HyperText Markup Language
<b>ICT</b>	Information and Communication Technologies
<b>I/M</b>	Inspection and Maintenance
<b>JSON</b>	JavaScript Object Notation
<b>KPI</b>	Key Performance Indicator
<b>ML</b>	Machine Learning
<b>NOx</b>	Nitrogen oxides
<b>NRMM</b>	Non-Road Mobile Machinery
<b>OBD</b>	On-Board Diagnostics
<b>OS</b>	Operating System
<b>PCA</b>	Principal Component Analysis
<b>PID</b>	Parameter ID
<b>RDBMS</b>	Relational Database Management System
<b>REST</b>	Representational State Transfer
<b>RPM</b>	Revolutions Per Minute
<b>SDK</b>	Software development kit
<b>t-SNE</b>	t-distributed Stochastic Neighbour Embedding
<b>URL</b>	Uniform Resource Locator
<b>WCAG</b>	Web Content Accessibility Guidelines
<b>WP</b>	Work Package
<b>XML</b>	Extensible Mark-up Language

## Executive Summary

This report is a support document to deliverable D5.3, which is a set of software solutions developed to collect data and evaluate the project's low emission driving innovations at trial sites in Europe. It is part of Work Package 5 of the MODALES project, entitled "Guidelines & tools for low emission training". D5.3, as a software deliverable, is key for MODALES as it integrates part of the results accumulated from more modelling tasks in WP2, WP3 and WP4, to serve as a basis for trial sites and data collection.

In this context, the objective of this report is to document how the software solutions of the project have been developed. It complements the specifications defined in the first phase of the project (D5.2: Functional Specifications) and serves as a reference for understanding the functioning of the different modules developed and their deployment and maintenance in the future.

The main solution that is described is the project's mobile application, which collects data from an On-Board Diagnostics (OBD) reader fitted to a vehicle and Smartphone sensors in addition to providing recommendations to end users (vehicle drivers). This application comes with support services, including:

- A reporting system to allow data storage and management within the consortium,
- An external web dashboard to display anonymised statistics to external stakeholders (e.g., public authorities) to help them monitor and understand the vehicle emissions in their region and thus develop effective policy interventions to reduce the emissions.

Interested readers can also refer to D5.4 (Experimental Test Results and Initial Feedback on User Acceptance), which includes further testing and details on the performance and components of these solutions.



# 1. Introduction

## 1.1. Background

The MODALES project works towards reducing air pollution from all types of on-road as well as Non-Road Mobile Machinery (NRMM) vehicles by encouraging adoption of low-emission driving behaviour and proper maintenance choice.

MODALES pursues a user-centric approach to addressing all the challenges which on the one hand enhance low-emission practices and on the other hand suppress high-emission behaviour by researching, developing, and testing several innovative and complementary solutions in four key areas (driver, retrofits, EOBD and inspection) in order to reduce vehicle emissions from three main sources: powertrain, brakes and tyres.

The scope of vehicles covered comprises passenger cars, light- and heavy-duty vehicles (buses and trucks) and NRMM.

The main activities of MODALES are:

- Measurement of real-world vehicle emissions and driving behaviour to produce accurate correlation between them using advanced mathematical and statistical techniques.
- Exploration of the most advanced technologies for retrofits designed to substantially reduce powertrain emissions from all types of vehicles and to validate their effectiveness under different real-world traffic and environment conditions, and by various drivers.
- Undertaking an in-depth analysis of OBD, periodic inspection and legal issues on tampering in Europe to help regulatory authorities put in place effective anti-tampering legislation, and to help owners properly maintain their vehicles.
- Conducting low-emission user trials (with both driving and maintenance practices), supported by awareness campaigns, to enhance public engagement and help drivers better understand the impact of their driving and maintenance behaviours in all situations.

## 1.2. Purpose and scope

This deliverable is part of Work Package 5 (WP5) on **Guidelines and Tools for Low-Emission Training**, which is one of the five technical WPs of MODALES (the two “non-technical” WPs include WP1 on Project Management and WP7 on Awareness, Communication and Dissemination). The other four “technical” WPs that are directly connected with WP5 are the following:

- **WP2: Defining low-emission factors**, which explored driving behaviour variability using existing available data. This WP delivered a first approach on driving behaviour patterns and powertrain, brake and tyre emissions. It also addressed the state-of-the-art in retrofits, inspection and maintenance (I/M) and legal issues regarding tampering in various EU Member States. [Link with WP5](#): outputs from WP2 used to develop the guidelines in WP5 (see Deliverable D5.1) and therefore the mobile app and the low-emission training developed in this Work Package.
- **WP3: Impact of user behaviours**, which undertook a series of measurement campaigns to establish the interconnection between driving behaviour and powertrain exhaust emissions, as well as fine particulates from brakes and mass-loss from tyres. Measurement campaigns were also carried out to address the impact of poor maintenance and deliberate tampering of the emissions control

system. Link with WP5: outputs from WP3 (reported in D3.2) to develop the guidelines and implement part of the profiling methods in WP5.

- **WP4: Effectiveness of inspections and depollution systems**, which used the findings of WPs 2 and 3 as a basis to investigate and propose solutions that will contribute to emission monitoring via the EOBD protocol and systems that detect lack of maintenance and tampering. It also investigated the potential of enhancing existing retrofit systems. Link with WP5: outputs from OBD work (reported in D4.1) to WP5 in order to integrate all the necessary data into the mobile app.
- **WP6: User trials and evaluation**, developed an evaluation plan and to test and evaluate with real-world trials the functionality of the innovations developed in MODALES, their effects on driver acceptance and performance, and their potential wider impact (their predicted overall effects on vehicle emissions). Link with WP5: the mobile app is used as the main tool for on-road user trials in WP6.

### 1.3. MODALES WP5 on low-emission driving tools

The first mission of WP5 is to ensure the link between the theoretical aspects of the project (WP2, WP3, WP4) and their validation and use for experimentation (WP6) and awareness campaigns (WP7). To do so, WP5 considers the results of all the WPs mentioned above, with the aim of (a) defining guidelines for low-emission driving and (b) defining tools to monitor and improve driving behaviour resulting in emission reductions. In this context, a mobile app has been developed as a driving assistance tool for low-emission driving. It is complemented by a web dashboard addressed to public and local authorities, providing them with a statistical overview of the data collected in WP6. This mobile assistant is developed and tested in this WP. Finally, the guidelines have been used as input for online training courses (videos), which were designed to ensure a consistency with existing learning processes and serve as input for on-road trials and awareness campaigns.

WP5 is broken down into five tasks:

- **Task 5.1 Guidelines for low-emission driving.** Objective: create a reference list of good and bad practices for low-emission driving, their impact and the actions the user could take to improve them.
- **Task 5.2 Functional specification of tools.** Objective: specify the tools to be developed in T5.3, including the mobile application and its modules, its operating logic, HMI aspects, the centralised web application.
- **Task 5.3 Low-emission driving assistance tools.** Objective: develop all the elements specified in T5.2 into advanced prototypes.
- **Task 5.4 Testing and technical verification of tools.** Objective: test, verify and deploy all the software developed and integrated during the WP. This Task also calibrates and technically validates the application.
- **Task 5.5 Developing training for low-emission driving.** Objective: use the guidelines, tools and all the knowledge accumulated in WP5 as an input to design the training courses carried out in WP6. Online training material is derived from the guidelines in T5.1, aimed at various types of drivers (including junior drivers, taxi drivers, driving schools, etc.) and for private car and professional/commercial vehicle drivers.

#### 1.4. Scope and intended audience of this deliverable

The content of this deliverable is public.

#### 1.5. Deviations from the Description of Action

The tools for low emission driving developed are in line with the original tasks foreseen in the DoA. However, the following must be pointed out:

- Graphical user interfaces using gamification concepts were initially planned. However, the consortium preferred a simple and non-disruptive display for the user, especially during the driving stages.
- The internal reporting system was not planned in the DoA and was developed to facilitate data management within the project, as described later in Section 2.
- A considerable number of challenges were encountered during the development of the app, especially regarding access to some OBD PIDs (Parameter IDs), which were not anticipated at the beginning of the project. These challenges and associated solutions/mitigation actions are described in D5.4.
- This deliverable suffered from a significant delay since, in addition to the points mentioned above, the integration of the models from WP3 and WP4 took much longer than expected, due to the already existing delay in WP3 and WP4 – cascading effect. The testing and deployment activities were also significantly delayed, in particular due to COVID-19.

## 2. Overview of MODALES tools

### 2.1. The mobile assistant

MODALES' mobile app for low-emission driving aims to assist users throughout their journeys, providing recommendations as they travel and in ways that do not interfere with their driving (e.g., when stationary at a traffic light or in the form of sound notifications).

Technically speaking, the app adopts a modular approach, which allows several small components to be developed independently and then interact with each other to provide a unique driver experience. This approach facilitates collaborative development and validation. The app has been developed for Android and iOS, using the Flutter framework to avoid working on two code bases. Flutter is Google's SDK for crafting beautiful, fast user experiences for mobile, web and desktop from a single codebase. In order to prevent breaches of privacy, most of the processes operate locally on the mobile phone. Furthermore, the local computation allows the system to operate even without Internet connectivity.

The app is broken down into three separate modules:

- A **data collection module**, which considers data from (a) OBD-II (e.g., engine rpm, vehicle speed), (b) phone sensors (e.g., accelerometer, wireless traces) and (c) the user. Using an OBD dongle is optional so as to provide end-users with a standalone application and another relying on additional data. The lack of data induced by the absence of OBD leads to less data collected for the project's impact analysis part, but still valid. Phones should, if possible, be fixed in the car using a car holder, thus allowing the user to have a comfortable view of the phone's interface and for the proper recording of accelerometric data, i.e., using a stable reference in space. Ideally, the phones should be positioned vertically, but the system also works horizontally.
- The data mentioned above is used as input to two **scoring modules**, making it possible to create a local representation of the user's profile and distinguish different driving behaviours previously identified via laboratory tests and state-of-the-art reviews. These scoring modules can compute, on one side, a real-time acceleration profile, and on the other side, a time series of scores, using a classification approach. Input data comes from OBD dongle and phone sensors, while ground truth data would be provided by the user and independent data such as GPS.
- Finally, a **recommendation module** advises the user to adopt attitudes depending on his/her perceived driving behaviour. Based on the two sub-scores explained above, the mobile app generates two types of recommendations:
  - **Active recommendations**, which are presented to the user while driving. The user's driving style is analysed in real-time and transcribed into an acceleration score. Active recommendations require local data processing and storage. The user interface and possible interactions actions needs to be limited to preserve the user's safety. These recommendations are kept simple.
  - **Passive recommendations**, which are given to the user after a trip. A report is generated, including textual recommendations, statistical analyses and contextual elements. This feedback allows drivers to learn from bad driving habits and improve their emissions reduction. Passive recommendations mean that data processing and storage will be outsourced to an external server.

The mobile app has been developed in two different versions: the first only for data collection (first module as shown in Figure 1), and the second with the full scoring and recommendation systems.

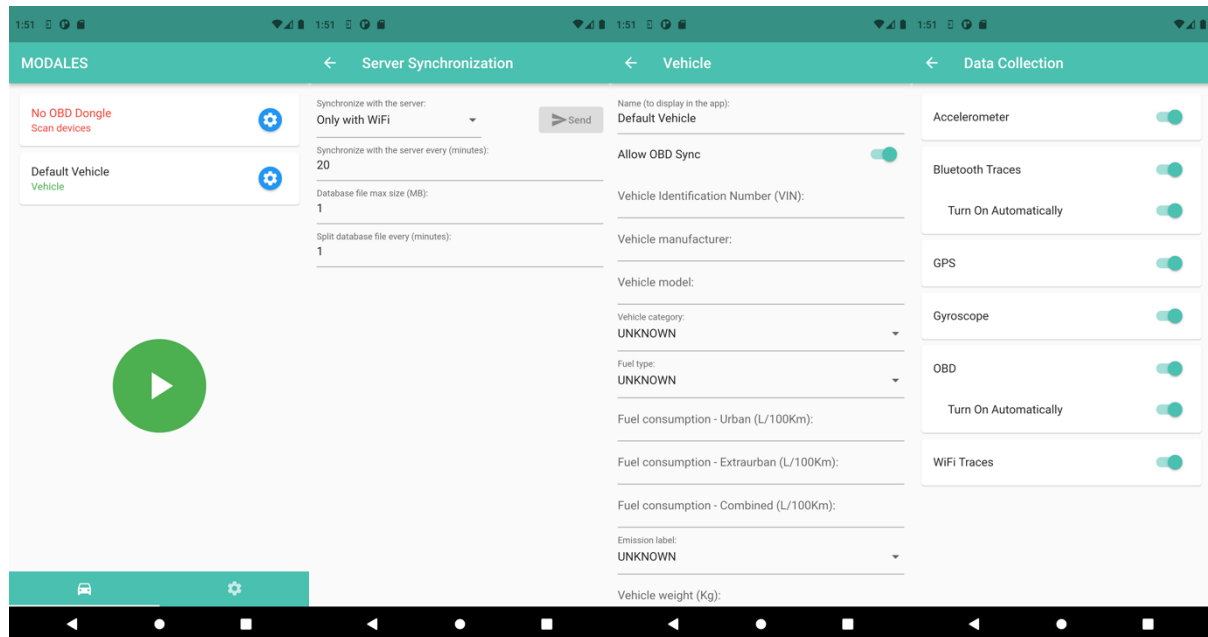


Figure 1: Screenshots of the app's data collection module

## 2.2. The project's dashboard

For each app user, anonymous indicators are transmitted to a **web dashboard** to collect usage statistics and performance metrics. The latter allows the authorities and potentially the public to understand the benefits of the mobile app and view statistics by region or type of user. The web dashboard aggregates the data and presents them in various graphical representations to assist decision-making.

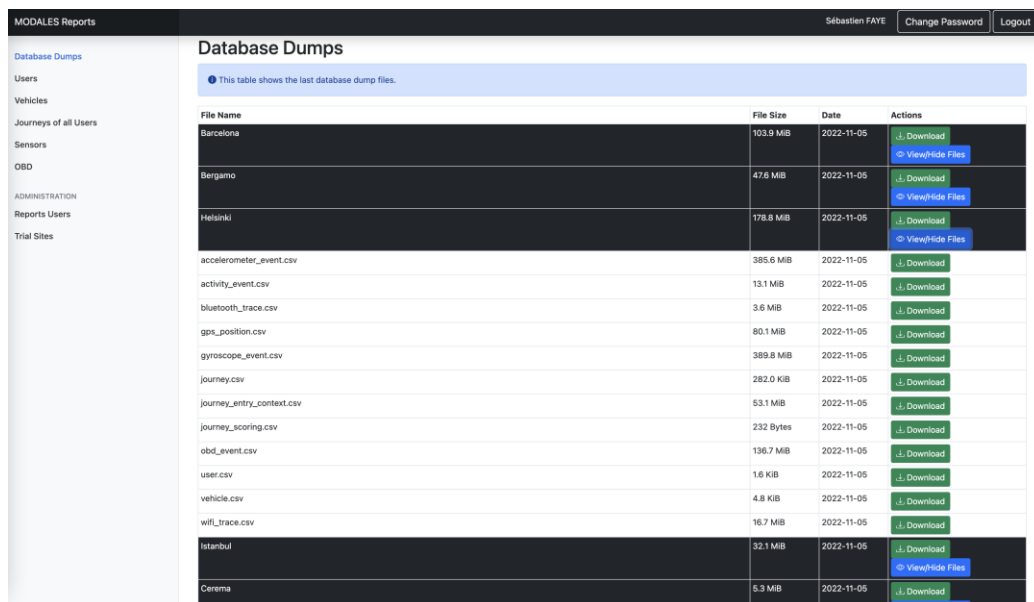


Figure 2: Dashboard web application

The web dashboard application is based on the received data from the mobile sensors and OBD dongles and indicators derived from them. The indicators include vehicle emissions, fuel consumption and driver's low emission performance. The three indicators are time-based. Thus, the dashboard can also present the performance evaluation based on time.

## 2.3. The internal reporting platform for trial site and data management

The data collected via the mobile app are, after post-processing on the LIST servers, stored in a PostgreSQL relational database. This data format is suitable for storage, but sharing it with trial site leaders and project partners managing the impact assessment activities is not ideal. Further, it is necessary to filter the data by trial site and to follow, in a simple and graphical way, the evolution of the data collection and the participation for each trial site. It is with this objective in mind that the reporting platform, hosted by LIST, was created. In addition to displaying statistical information, as illustrated in the screenshots below, it allows the automatic availability (every night) of the data sets for each trial site.



File Name	File Size	Date	Actions
Barcelona	103.9 MiB	2022-11-05	<a href="#">Download</a> <a href="#">View/Hide Files</a>
Bergamo	47.6 MiB	2022-11-05	<a href="#">Download</a> <a href="#">View/Hide Files</a>
Helsinki	178.8 MiB	2022-11-05	<a href="#">Download</a> <a href="#">View/Hide Files</a>
accelerometer_event.csv	385.6 MiB	2022-11-05	<a href="#">Download</a>
activity_event.csv	13.1 MiB	2022-11-05	<a href="#">Download</a>
bluetooth_trace.csv	3.6 MiB	2022-11-05	<a href="#">Download</a>
gps_position.csv	80.1 MiB	2022-11-05	<a href="#">Download</a>
gyroscope_event.csv	389.8 MiB	2022-11-05	<a href="#">Download</a>
journey.csv	282.0 KiB	2022-11-05	<a href="#">Download</a>
journey_entry_context.csv	53.1 MiB	2022-11-05	<a href="#">Download</a>
journey_scoring.csv	232 Bytes	2022-11-05	<a href="#">Download</a>
obd_event.csv	136.7 MiB	2022-11-05	<a href="#">Download</a>
user.csv	1.6 KiB	2022-11-05	<a href="#">Download</a>
vehicle.csv	4.8 KiB	2022-11-05	<a href="#">Download</a>
wifi_trace.csv	16.7 MiB	2022-11-05	<a href="#">Download</a>
Istanbul	32.1 MiB	2022-11-05	<a href="#">Download</a> <a href="#">View/Hide Files</a>
Canema	5.3 MiB	2022-11-05	<a href="#">Download</a> <a href="#">View/Hide Files</a>

Figure 3: Reporting platform: dataset download for each trial site

MODALES Reports

Sébastien FAYE

Change Password

Logout

Database Dumps

Users

Vehicles

Journeys of all Users

Sensors

OBD

ADMINISTRATION

Reports Users

Trial Sites

OBD

This table shows which OBD requests were answered for each vehicle. Note that some users may have more than one vehicle.

References:

ECT: engine coolant temperature

ES: engine speed

VS: vehicle speed

ABP: absolute barometric pressure

CT: catalyst temperature

AAT: ambient air temperature

ATP: absolute throttle position

APP: accelerator pedal position

EFR: engine fuel rate

MAF: MAF (mass air flow sensor) air flow rate

IAT: intake air temperature

NOX: nitrogen oxide

BX: bank X

SX: sensor X

User ID	ECT	ES	VS	ABP	CT				AAT	ATP	APP					EFR	MAF	IAT					NOX				Actions						
					B1		B2				B	C	D	E	F			B1			B2		B1	B2	S1	S2							
					S1	S2	S1	S2										S1	S2	S3	S1	S2						S3	S1	S2	S1	S2	
BCN10009	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙		
BCN10016	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙		
BCN10016	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	
BCN10016	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	
BCN10016	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	
BCN10016	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙

Figure 4: Reporting platform: OBD data collection analysis

This platform comes with a multi-rights management of users, as well as the possibility to define new trial sites. As defined in the project's data management plan, users are affiliated to a trial site according to their ID, provided by the trial site leaders when they register.

## 2.4. Overview of development responsibilities

The following table shows an overview of the different software components that were developed and the partners that were responsible for their implementation and maintenance.

**Table 1: Overview of the software components developed in the project and leadership**

Component		Leader (MODALES partner)
<b>Mobile app</b>	Data collection	LIST
	Active recommendation	LIST
	Passive recommendation	OKAN
<b>Reporting platform</b>		LIST
<b>Web dashboard</b>		CERTH

## 3. Technical implementation

### 3.1. Specifications

The specifications of the low-emission driving tools, as originally foreseen in the DoA, were described in Deliverable D5.2 – Functional specifications. This deliverable includes use-cases, mock-ups of graphical interfaces, development timelines, architecture designs, analysis of required technologies, data definitions, and the connection with the various theoretical aspects defined in previous deliverables.

### 3.2. High-level architecture

The figure below shows the overall architecture of the mobile app, which is the core of the software solutions developed within MODALES. The different components of the architecture are detailed in the following sections.

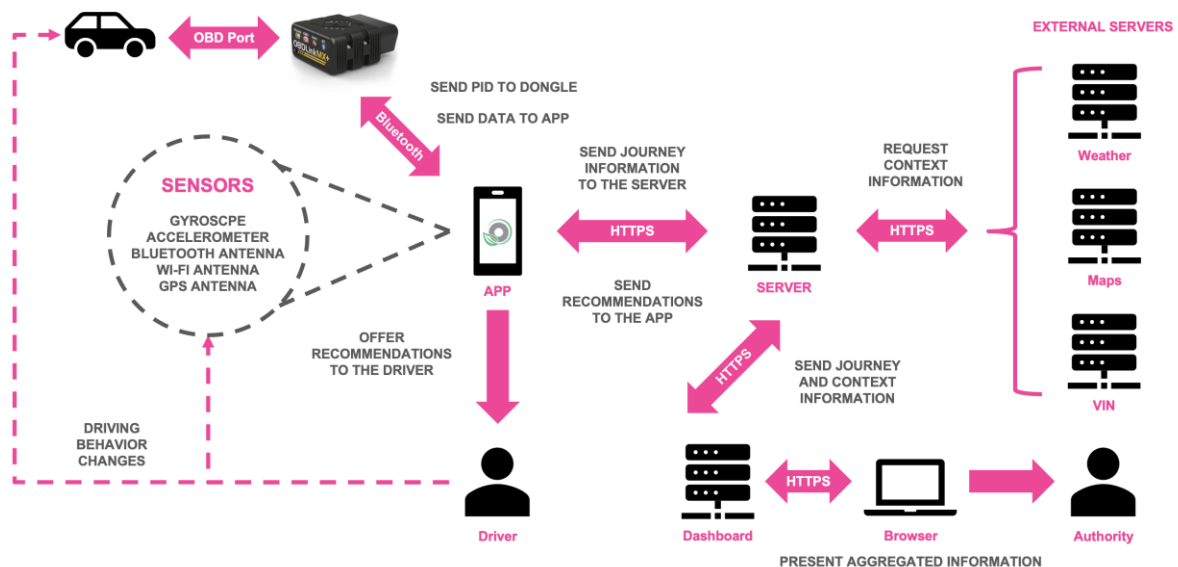


Figure 5: High-level architecture of the mobile app

Two main parts are described: the client side (the mobile app itself) and the server side (a mix of several components to collect, process, extend and store the data received from the app). Optionally, there is an OBD dongle that is connected to the vehicle using the OBD port and connected to the mobile device via Bluetooth.

### 3.3. OBD data collection

Part of the data collection was done directly from OBD dongles connected via Bluetooth to the smartphone, and selected following the study carried out in D4.1. The development operations were done in laboratory in a first step using an OBD emulator, reproducing messages in a realistic way in order to predict the behaviour of the mobile application and program OBD requests.

- Dongle: [OBDLINK® MX+](#)
- Emulator:
  - Software: [Freematics OBD-II Emulator GUI](#)



- Hardware: Freematics OBD-II Emulator



Figure 6: Illustration of the link between the car and the mobile app

- **More reliable data collected:**

- Absolute barometric pressure
- Absolute throttle position
- Accelerator pedal position
- Air flow rate
- Ambient air temperature
- Catalyst temperature
- Engine coolant temperature
- Engine fuel rate
- Engine speed
- Vehicle speed

- **Less reliable data collected (i.e., not always available in the cars):**

- Intake air temperature
- NOx sensor

### 3.4. Client Side

The client side of the mobile application is developed in Flutter, with some platform specific submodules. It follows the specifications already described in D5.2, and consists of the functionalities described below.

- **Profile:**

- Internal Unique User Identifier (UUID).
- Manually inputted User Identifier (ID).
- Driver's license information.

- **Session:**

- When opening the application for the first time, it creates an empty profile.
- It reads and writes the profile from and to the disk using a secure key-value store.

- **Preferences:**

- When opening the application for the first time, it defines default preference values.
- It reads and writes the preferences from and to the disk using a key-value store.

- **Databases:**

- Sensors: stores entries that are deleted after sync.
- Static: stores entries that are no deleted after sync.

- **Journey:** manages the journey status.
- **Sensors:** modules that capture and log sensor information into the local database.
  - Accelerometer: listens to accelerometer events.
  - Activity: listens to activity events.
  - Aggregator: thread that aggregates all the sensor information periodically.
  - Bluetooth: thread that scans beacons periodically.
  - GPS: listens to GPS events.
  - Gyroscope: listens to gyroscope events.
  - OBD: thread that sends requests to the OBD dongle periodically.
  - Wi-Fi: thread that scans access points periodically.
- **OBD:** module that handles all the OBD related actions.
  - Command: defines the command payload to be sent and the response parser.
    - AT: administrative command.
    - Value Request: asks the value based on Service and PID.
    - Multi Value Request: encapsulates several value requests.
  - Response Parser: given a response payload, executes a success callback with the parsed value or an error callback with a message.
    - AT Response Parser: parses responses for administrative commands.
      - Check response is OK: validates if payload == "OK".
      - Ignore response: does not take any action given a payload.
    - Value Response Parser: it contains all the common validation for a response containing a value, and if everything is right, it delegates the actual parsing of the received value to a simpler module called "Value Parser".
    - Multi Value Response Parser: expects multiple value responses from a single command, but they can arrive in arbitrary orders.
  - Value Parser: transforms response bytes into a specific response value ([reference](#)).
    - Fuel type: maps from 23 fuel type IDs into our 8 arbitrary fuel types.
    - IAT: parses Intake Air Temperature sensor values.
    - NOx: parses Nitrogen Oxides sensor values.
    - Scale Offset: value = byte / scale + offset.
    - Percent: value = byte \* 100 / 255.
    - VIN: parses a Vehicle Identification Number.
  - Response Value: represents a complex response value (i.e., not a float).
    - Intake Air Temperature Sensor Values: values for 3 sensors in 2 banks.
    - NOx Sensor Values: values for 2 sensors in 2 banks.
    - OBD PIDs Supported Mask: given an offset, the parsed byte indicates which PIDs are accepted by the vehicle in the range [offset, offset + 32].
  - Device: contains the dongle address and name.
    - Android: indicates if the dongle is connected and paired.
    - iOS: only indicates if the dongle session started.
  - Device Whitelist: indicates if a device name is accepted.
  - Device Manager:
    - Responsibilities:
      - Connection and Disconnection:
        - Delegate the low-level logic to the platform specific code.

- Handle common connection state logic.
- Message sending and Response reception:
  - Delegate the low-level logic to the platform specific code.
  - Delegate the high-level logic to the interpreter.
- Configuration:
  - Set flags and establish supported PIDs.
  - Executed right after connection.
  - Depends on message sending and response reception.
- Platform specific code:
  - Android:
    - Based on “Flutter Bluetooth Serial” plugin.
    - Bluetooth status can be verified and manipulated.
    - Bluetooth scans can be managed.
    - Bluetooth devices can be paired and unpaired.
  - iOS:
    - Based on “Apple Accessories API”.
    - Coded in swift by us.
    - We can launch the accessory window.
    - We cannot scan, pair or unpair accessories.
    - We can ask for the list of “paired” accessories.
    - We can start or end a session with a “paired” accessory.
- Interpreter: maintains the state of the conversation with the dongle.
  - Serial: the dongle protocol only allows one command at a time.
  - Asynchronous: modelled with dart “Futures” (like JavaScript “Promises”).
  - Responsibilities: perform certain actions or react to certain events.
    - Encode command: transforms a command into bytes to be sent.
    - Next command: prepares the internal status for a new command.
    - Command could not be sent: the current command did not reach the dongle and the status needs to be cleared in the interpreter.
    - Connection lost: needs to clear the status and notify accordingly.
    - Decode response: derived to internal events.
      - Prompt received: when the dongle indicates that it is waiting for new commands.
      - Response received: when the payload of a response arrives.
      - Error received: when the current command failed.
  - States: the expected execution path will be described below. However, in case something unexpected happens, each state has the corresponding logic to handle every alternative path.
    - Idle: the dongle is ready for a new command.
    - Waiting for confirmation: the command was sent to the dongle. In case the echo is on, we should receive back the same command as an acknowledgement. If echo is off, this state is not used.
    - Waiting for response: the command was sent, and the interpreter is ready to receive and parse the response.
    - Waiting for prompt: the response was received and parsed. When the prompt is received, the state cycle will finish calling a success callback with the response contents or an error callback with a message.

- **Server:** takes care of the communication with our REST API.
- **Synchronisation:**
  - Listens to connection status events.
  - Uses the preferences module to define which connection type to use.
  - When there is a connection, it launches a periodic synchronisation thread.
  - Uses the server module to send data to our REST API.
  - Deletes local information that was synchronised.
- **Widgets:** entire screens or smaller portions of the UI.
  - Pre-load: loads resources asynchronously while showing a spinning wheel.
  - Tabs: main screen with two tabs.
    - Journey: current trip tab.
      - Play/Stop button: controls the status of the journey.
      - Vehicle window: shows the selected vehicle.
      - OBD dongle window: shows the OBD dongle connection status.
    - Settings: a list with access to other setting screens.
      - Profile: user form.
      - OBD Dongles: list, scan, pair, unpair and connect to dongles.
      - Garage:
        - List, add, remove, and edit vehicles.
        - Select current vehicle.
      - Data Collection: change data collection parameters.
      - Server Synchronisation:
        - Change server synchronisation parameters.
        - Synchronise with the server manually.
      - Clear Stored Data:
        - Deletes the sqlite databases.
        - Restores the preferences to the default values.
        - It does not remove anything from the server.

### 3.5. Server Side

The server is directly linked to the mobile application and is hosted by LIST. It implements all the logic to collect, pre-process and store the data. It also provides the APIs and services to deliver the data to the partners and the web dashboard developed by CERTH. The different modules implemented are detailed below.

- **Apache2:** serves a virtual site using SSL (port 443). Inside the virtual site there are two paths running two different `mod_wsgi` applications developed with flask:
  - `/api`: MODALES API to communicate with mobile devices.
    - `/user`: create or update a user
    - `/vehicle`: create or update a vehicle
    - `/sensor_logs`: dump sensor sqlite file
    - `/system_logs`: dump log zip file
  - `/reports`: MODALES reports website for our partners.
    - `/users`: list users
    - `/vehicles`: list vehicles

- `/journeys`: list journey status per vehicle
- `/sensors`: list sensor availability per vehicle
- `/obd`: list OBD PID availability per vehicle
- `/trial_sites`: (Administrator only) manage trial sites for the report website
- `/reports_users`: (Administrator only) manage users for the report website
- **phpPgAdmin**: to make database administration easier (through SSH tunnel on port 8080).
- **PostgreSQL**: stores data for the API and the reports website at the same time.
- **Redis**: used as a queue for the async tasks (see Celery).
- **Elasticsearch**: just stores log data (currently only from the mobile devices).
  - Proxy configuration defined in `/etc/default/elasticsearch`
- **Kibana**: only used to query and visualise logs (through SSH tunnel on port 5601).
  - Proxy configuration defined in `/etc/default/kibana`
  - Proxy configuration to install integrations in `/etc/kibana/kibana.yml`
- **Celery**: to manage and execute async tasks.
- **Flower**: visualise Celery tasks (through SSH tunnel on port 5555).
- **Docker**: installed mostly to run the OKAN University Scoring container.
  - Proxy configuration defined in `/etc/default/docker`

### 3.6. External Services

External services are used by the server to augment the data collected via the mobile application, as detailed below.

- **Motion-S**:
  - Gives context information (weather, road, and traffic conditions) based on GPS.
  - [Company website](#)
  - [Documentation page](#)
  - Data interpreted from GPS (see Section 3.8.1 for more details):
    - **Weather**:
      - humidity
      - temperature
      - total snow
      - visibility
      - wind speed
    - **Traffic**:
      - historical average speed
      - historical free flow speed
      - historical jam factor
      - historical traversability
      - real-time confidence
      - real-time traffic speed
    - **Road**:
      - average roughness category
      - curvature

- distance to intersection
- functional class
- has end of no overtaking sign
- has no overtaking sign
- has pedestrian crossing sign
- has stop sign
- has traffic signal
- has yield sign
- heading
- intersection category
- international roughness index
- is bridge
- is intersection
- is long haul
- is ramp
- is roundabout
- is tunnel
- is urban
- lane category
- radius
- road type
- slope
- speed category
- speed limit
- speed limit trucks

- **VIN Decoder:**

- Gives vehicle information based on a VIN.
- [Company website](#)
- [Documentation page](#)

- **OKAN University Scoring** (more details in Section 5).

- Gives emission scores based on speed and distance time series.
- Executed in our server as a docker container:  

```
docker run -d --name=okan-scoring -p 8102:8102 eozatay/scoring:1.2.0
```

### 3.7. Deployment

The deployment environment of the two versions of the application (Android, iOS) and the server are detailed below.

- **Android:**

- Download link: [Google Play](#)
- Requirements: Android 11 (API level 30) or greater.
- Permissions:
  - Closed testing.
  - Each trial site created a Google Group.

- Each Google Group email was added by us to the tester list.
- Volunteers are added to each Google Group.

- **iOS:**

- Download link: [TestFlight](#)
- Requirements: iOS 9 or greater.
- Permissions:
  - We did not want to manage all the volunteers at LIST.
  - There is no option like the Google Groups counterpart.
  - Our only option was to open the testing for anyone with the link.

- **Servers:**

- Development: private URL
- Production:
  - Public, hidden URL for security purposes

## 3.8. Databases

### 3.8.1. Server side

The server stores the data using a relational database, the structure of which is detailed in the figure below.

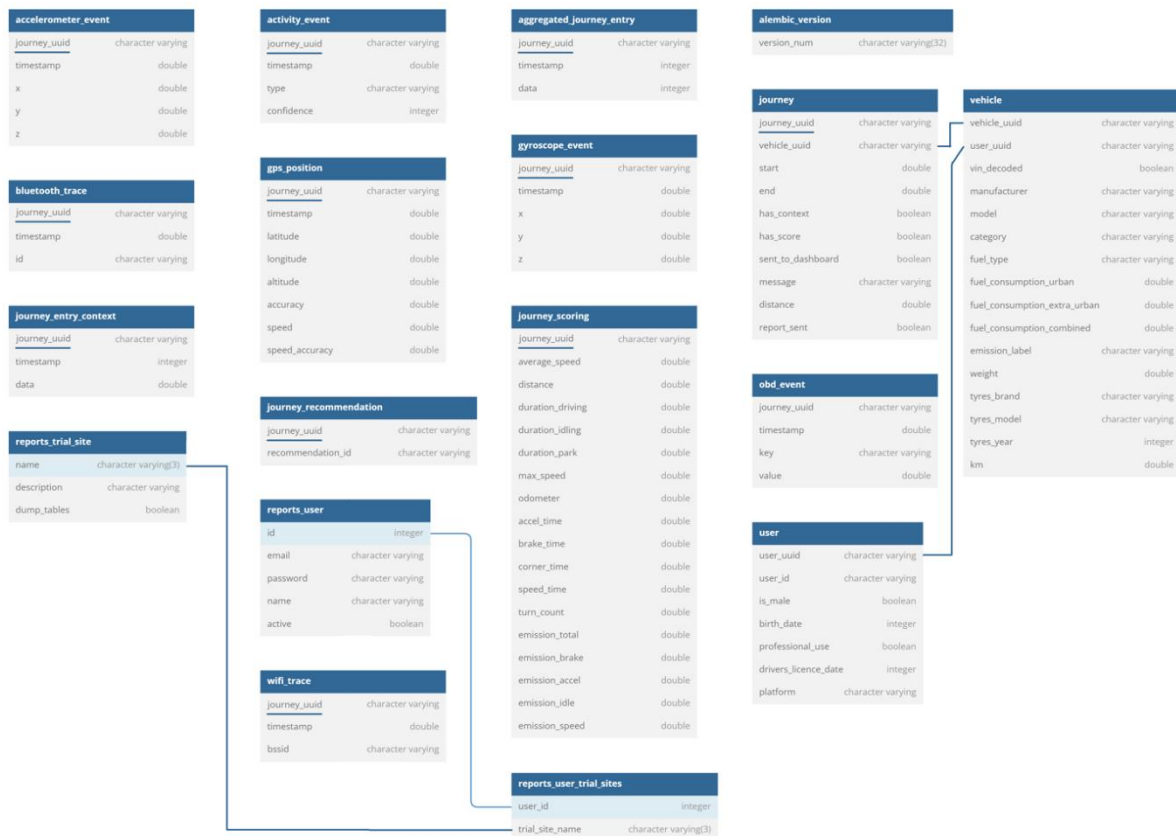


Figure 7: Database structure of the MODALES server

### 3.8.2. Client side

The database on each phone is just like the database on the server. However, the server contains additional tables to store information about context, scoring, and the report website.

- **journey:** created every time a journey is detected
  - journey\_uuid
  - vehicle\_uuid
  - start
  - end
- **activity\_event:** written by the Activity Event module and read by the Aggregator module
  - journey\_uuid: link to the ongoing journey when the measurement was taken
  - timestamp: time when the measurement was taken
  - type: string defining the activity
    - IN\_VEHICLE
    - ON\_BICYCLE
    - ON\_FOOT
    - RUNNING
    - STILL
    - TILTING
    - UNKNOWN
    - WALKING
    - INVALID
  - confidence: how confident is the module about the prediction (0 to 100)
- **accelerometer\_event:** written by the Accelerometer module and read by the Aggregator
  - journey\_uuid: link to the ongoing journey when the measurement was taken
  - timestamp: time when the measurement was taken
  - x
  - y
  - z
- **bluetooth\_trace:** written by the Bluetooth module and read by the Aggregator module
  - journey\_uuid: link to the ongoing journey when the measurement was taken
  - timestamp: time when the measurement was taken
  - id
- **gyroscope\_event:** written by the Gyroscope module and read by the Aggregator module
  - journey\_uuid: link to the ongoing journey when the measurement was taken
  - timestamp: time when the measurement was taken
  - x
  - y
  - z
- **gps\_position:** written by the GPS module and read by the Aggregator module
  - journey\_uuid: link to the ongoing journey when the measurement was taken



- timestamp: time when the measurement was taken
- latitude
- longitude
- altitude
- accuracy
- speed
- speed\_accuracy
- **obd\_event**: written by the OBD module and read by the Aggregator module
  - journey\_uuid: link to the ongoing journey when the measurement was taken
  - timestamp: time when the OBD requests started
  - duration: amount of time that was needed to finish all the OBD requests
  - engine\_coolant\_temperature
  - engine\_speed
  - vehicle\_speed
  - maf\_air\_flow\_rate
  - absolute\_barometric\_pressure
  - catalyst\_temperature\_bank1\_sensor1
  - catalyst\_temperature\_bank2\_sensor1
  - catalyst\_temperature\_bank1\_sensor2
  - catalyst\_temperature\_bank2\_sensor2
  - ambient\_air\_temperature
  - absolute\_throttle\_position\_b
  - absolute\_throttle\_position\_c
  - accelerator\_pedal\_position\_d
  - accelerator\_pedal\_position\_e
  - accelerator\_pedal\_position\_f
  - engine\_fuel\_rate
  - intake\_air\_temperature\_bank1\_sensor1
  - intake\_air\_temperature\_bank2\_sensor1
  - intake\_air\_temperature\_bank1\_sensor2
  - intake\_air\_temperature\_bank2\_sensor2
  - intake\_air\_temperature\_bank1\_sensor3
  - intake\_air\_temperature\_bank2\_sensor3
  - nox\_bank1\_sensor1
  - nox\_bank2\_sensor1
  - nox\_bank1\_sensor2
  - nox\_bank2\_sensor2
- **wifi\_trace**: written by the Wi-Fi module and read by the Aggregator module
  - journey\_uuid: link to the ongoing journey when the measurement was taken
  - timestamp: time when the measurement was taken
  - bssid: identifies the discovered access point

## 4. Focus on: the active recommendation system

### 4.1. Instantaneous user acceleration

The main purpose of active recommendation is to make users react when a bad driving behaviour is detected. Several approaches were considered, including gamification, complex interfaces, audio or text announcements, etc. In the end, we adopted a simple, non-distracting approach for the driver, based on a colour code.

To do this, a score is calculated on the phone, in real time and with a very dynamic refresh rate so that the user can immediately see the effects of his/her driving on the system. It was decided that this score should allow the detection of situations where the driver would or would not have smooth or more aggressive driving behaviour (Deliverable D5.2).

It was initially thought that this score considers several metrics, such as the revolutions per minute (RPM), collected via OBD, or GPS speed, to calculate an acceleration/deceleration coefficient. The problem in both cases is that the collection frequency is usually more than one second, which prevents the application and the interface from being as responsive as possible. Furthermore, we did not want to create dependencies on external services or devices and avoid energy-greedy solutions. This score should be calculated locally on the phone and without external influence, even when offline.

The solution that was adopted was to consider the accelerometer incorporated in all smartphones on the market, and measuring the force experienced by the device on three axes (x, y, z). This sensor is permanently used by smartphones to detect the user's activity and consumes very little energy. Moreover, the sampling frequency is usually 100-200 milliseconds which is perfect in our case.

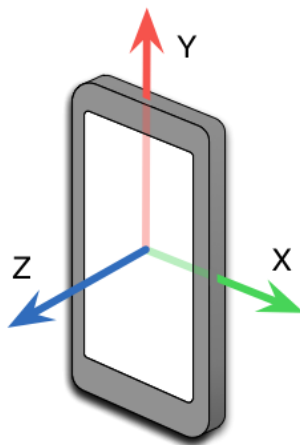


Figure 8: Smart axes (source: medium.com)

The calculated score is equal to the acceleration of the device on the z axis, without considering the gravity. The linear acceleration has also been considered, but was too sensitive to the user's movements:

$$\text{Linear acceleration} = \sqrt{x^2 + y^2 + z^2}$$

This acceleration is quite precise, but with the obligation to have the smartphone fixed on a support. Future versions of the app could consider using the linear acceleration to be agnostic from the position of the phone.

#### 4.2. Real-time recommendations

Active recommendations are also considered part of the training and behaviour change methodology. The objective of the real-time recommendation system is to provide an indication to the driver while driving, which means that it must carefully be designed so as not to distract the driver. For this reason, colouring has been selected with no text so that the driver will not be distracted.

As described in the table below, different threshold values have been used to decide about the colour showcased to the user. Mainly four colours are used: green, yellow, orange and red changing in relation to the severity of driving aggressiveness. These threshold values are based on the scientific literature<sup>1</sup>.

**Table 2: Speed Range vs Acceleration and Deceleration Thresholds**

0-10	10-20	20-30	30-40	40-50	50-60	60-70	70-80	80-90	> 90
<0,3	<0,3	<0,25	<0,25	<0,25	<0,20	<0,20	<0,20	<0,20	<0,15
0,3-0,4	0,3-0,4	0,25-0,35	0,25-0,35	0,25-0,30	0,20-0,30	0,20-0,30	0,20-0,30	0,20-0,30	0,15-0,25
0,4-0,5	0,4-0,5	0,35-0,45	0,35-0,45	0,30-0,40	0,30-0,40	0,30-0,40	0,30-0,40	0,30-0,40	0,25-0,35
>0,5	>0,5	>0,45	>0,45	>0,40	>0,40	>0,40	>0,40	>0,40	>0,35
Braking, values are negative (deceleration)									
0-25	25-60	60-100	>100						
<0,4	<0,25	<0,25	<0,2						
0,4-0,5	0,25-0,3	0,25-0,3	0,2-0,25						
>0,5	0,3-0,40	0,3-0,35	0,25-0,3						
	>0,4	>0,35	>0,3						

A weighting factor of 0.5 has been applied to these threshold values, in order to give more proactive feedback to the driver. This weighting factor has been determined empirically, based on the preliminary testing and data collection realised through WP6 (T6.2). These colours are then used to indicate to the user a recommendation level, as shown in the screenshots below.

<sup>1</sup> Kurtyka, Karolina, and Jacek Pielecha. "The evaluation of exhaust emission in RDE tests including dynamic driving conditions." *Transportation Research Procedia* 40 (2019): 338-345.

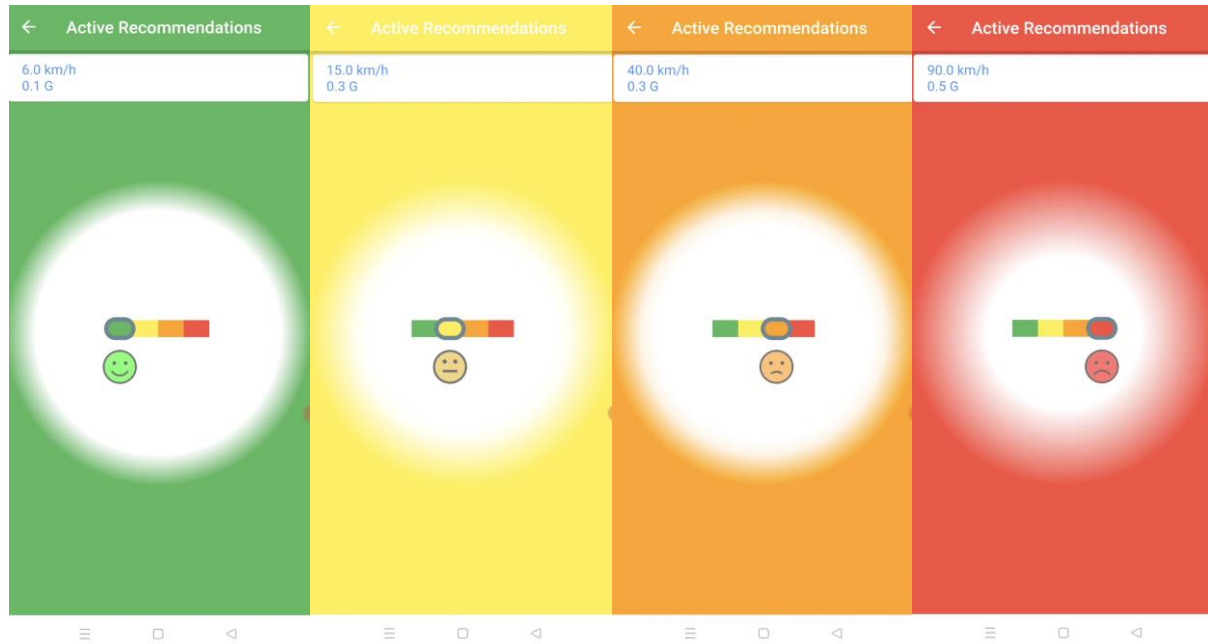


Figure 9: Active recommendation system

## 5. The passive recommendation system

### 5.1. Scoring system

A post trip scoring system will give the driver a realistic feedback about his/her driving profile as to optimise the total emissions. The score calculated must be related to the total emissions including the particles emitted from the brakes and tires. Below several aspects of the scoring methodology is described

#### 5.1.1. Scoring methodology outline

The methodology for the development of the scoring system has been explained in Deliverable 5.1 in Section 3.2. The literature survey has shown that for the detection of driving events with respect to the emissions v.a\_95 percentiles and RPA<sup>2</sup> gives reasonably good results as explained in Section 2.1.1 of the same deliverable. The scoring methodology as selected uses an AI system with random forest algorithm.

The first step is the determination of the events and the related features. The events are determined through v.a values detecting the accelerations greater than  $0,1 \text{ m/s}^2$  as indicated at Commission Regulation (EU) 2018/1832. Then calculations for RPA is carried out and the features for emissions are deduced. These features are input to random forest algorithm for training. Training leads to the score determining the threshold values for each velocity acceleration combination and the multiplier factors in relation to RPA values. The following diagram depicts the main flow chart for the algorithm.

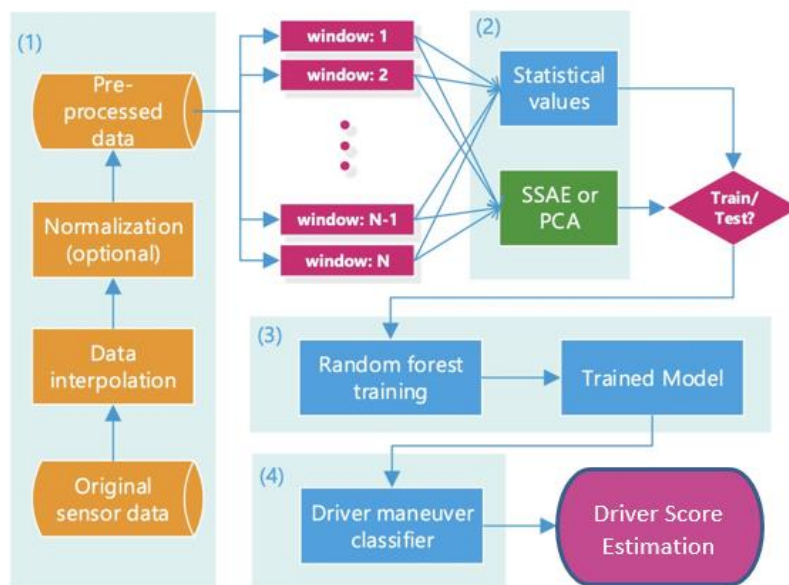


Figure 10: Scoring methodology for passive recommendations

This system is flexible, and it can be trained for any vehicle type with the condition that there is training data existing for that type of vehicle with precise emission values.

<sup>2</sup>RPA : relative positive acceleration for urban, rural and motorway driving [ $\text{m/s}^2$  or  $\text{kWs}/(\text{kg} \cdot \text{km})$ ]

v.a\_95 : 95th percentile of the product of vehicle speed and positive acceleration greater than  $0,1 \text{ m/s}^2$  for urban, rural and motorway driving [ $\text{m}^2/\text{s}^3$  or  $\text{W/kg}$ ]

### 5.1.2. Parametrised Multi-objective Training Set Preparation for aggregate emission

In Deliverable 3.1, the road tests conducted by partner VTT in Finland with six passenger cars and a standard PEMS device measuring the emission gases CO, CO<sub>2</sub>, NO, NO<sub>2</sub>, NO<sub>x</sub>, PN and fuel consumption. For each vehicle there are around 30 sets of trip data reaching to more than 180 sets of data. This data is sufficient for sound training; however the training tables must be prepared for aggregate emissions, including all the different types exhaust gases and particles. Different exhaust gases and particles have different impact on human health. As a first stage a parametrised multi-objective aggregation for the emissions and particles is to be prepared. Determination of the parameters have been carried out by using the “Air Quality Index”<sup>34</sup> and health costs related to the exhaust gases and particles effect on human health<sup>5</sup>. Also considering the fuel consumption causing carbon emissions accelerating global warming, the following weights for the parametrisation of different emissions have been determined to be used for the training sets.

Fuel consumption	NO <sub>x</sub>	PM <sub>10</sub>	PM <sub>2.5</sub>
0,50	0,09	0,14	0,27

On the training data sets VTT tables are reorganised by multiplying the fuel consumption and emissions with the above weights to find one multi-objective aggregate emission indicator. However the tables are to be redesigned once again by adding PM values coming from brake and tyre wear. The formulae for PM values calculations are given at deliverable 3.2 section 4 and 5 as follows,

For the brake wear (m),

$$m = \rho V_w = \varphi K \frac{F_N \rho L}{3} = \frac{K \rho \varphi M}{6 N k} (v_1^2 - v_2^2) \quad \left\{ \begin{array}{l} \varphi=1 \quad T_{\text{pad}} < 200^\circ\text{C} \\ \varphi=1.8 \quad 200^\circ\text{C} \leq T_{\text{pad}} \leq 250^\circ\text{C} \\ \varphi=5.6 \quad T_{\text{pad}} > 250^\circ\text{C} \end{array} \right.$$

Where,

- $N$  is the number of brake assembly of each vehicle;
- $k$  is friction coefficient;
- $M$  is mass of vehicle;
- $v_1$  is velocity when the vehicle begins to brake;
- $v_2$  is velocity when the vehicle stop to brake
- $\rho$  is density of brake pad or brake disc;
- $\varphi$  is coefficient regarding brake pad temperature

Tyre Wear formula is given at 5.5 of deliverable 3.2 as follows,

<sup>3</sup> DIRECTIVE 2008/50/EC OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL of 21 May 2008 on ambient air quality and cleaner air for Europe

<sup>4</sup> <https://www.eea.europa.eu/themes/air/air-quality-index>

<sup>5</sup> Monetary values of exhaust and non-exhaust emissions emitted from conventional and electric vehicles (Ye Liu, Haibo Chen, Ying Li b, Jianbing Gao, Kaushali Dave, Junyan Chen, Tiezhu Li, Ran Tu), Journal of Cleaner Production, Volume 331, 10 January 2022, 12996

$$m_T = \varphi k_1 (w)^{k_2} B D$$

$$w = \frac{P(t)}{\varphi N B L}$$

Where,

- $m_T$  is the mass loss;
- $\varphi$  is the transverse reduction coefficient due to the tyre pattern;
- $w$  is the frictional power per unit contact area;
- $k_1$  and  $k_2$  are two constants that characterise the wear behaviour of the rubber compound at a given temperature and on a given abrasive surface;
- $B$  is contact width between tyre and ground and  $D$  is the driven distance of vehicle;
- $N$  is the number of tyre of the vehicle;  $L$  is the contact length between tyre and ground;
- $P(t)$  is the energy consumption.

The  $P(t)$  can be obtained from the equation below according to the literature

$$P(t) = \frac{1}{3600\eta_d} v(t) \left\{ \frac{\rho}{25.92} C_D C_h A_f v(t)^2 + \frac{g M C_r}{1000} [c_1 v(t) + c_2] + g M G(t) + (1 + \lambda) M \frac{dv}{dt} \right\}$$

Where,

- $C_h$  is altitude;
- $g$  is gravity;
- $\rho$  is air density;
- $G(t)$  is slope;
- $M$  is vehicle weight;
- $A_f$  is vehicle frontal area;
- $C_D$  is aerodynamic drag coefficient;
- $C_r$  is rolling resistance factor;
- $c_1$  and  $c_2$  are rolling parameters;
- $\eta_d$  is driveline efficiency;
- $\lambda$  is Rotl masses. (usually 0.8 for passenger cars)

After the calculations to obtain tyre and brake particle mass, new columns for total  $PM_{10}$  and  $PM_{2.5}$  is necessary. Literature<sup>6</sup> shows that for Tyres  $PM_{10}$  and  $PM_{2.5}$  distribution is almost equal whereas for brakes 40% for  $PM_{10}$  and 60% for  $PM_{2.5}$  will be more adequate.

### 5.1.3. Threshold and multiplication values for score calculation

Training is used to fine-tune the threshold values for each speed range and also to determine the penalty values related to duration and extensions of acceleration over the threshold values. The threshold values determined for passenger cars are given below.

---

<sup>6</sup> "Brake Wear Particulate Matter Emissions", Garg et.al, September 2000 Environmental Science and Technology

Thres. (g)	Velocity (kph)												
	0	10	20	30	40	50	60	70	80	90	100	110	120
Acceleration	0.30	0.30	0.25	0.25	0.25	0.20	0.20	0.20	0.20	0.15	0.15	0.15	0.15
Deceleration	-0.50	-0.50	-0.50	-0.25	-0.25	-0.25	-0.25	-0.20	-0.20	-0.20	-0.20	-0.20	-0.20
Left Turn	0.35	0.35	0.35	0.35	0.30	0.30	0.30	0.25	0.25	0.25	0.20	0.20	0.20
Right Turn	-0.35	-0.35	-0.35	-0.35	-0.30	-0.30	-0.30	-0.25	-0.25	-0.25	-0.20	-0.20	-0.20

Figure 11: Threshold values determined for passenger cars

An example for the acceleration profile is shown below to explain the penalty factor in relation to the acceleration duration and value,

Sample Threshold & Penalty Coefficient Calculation

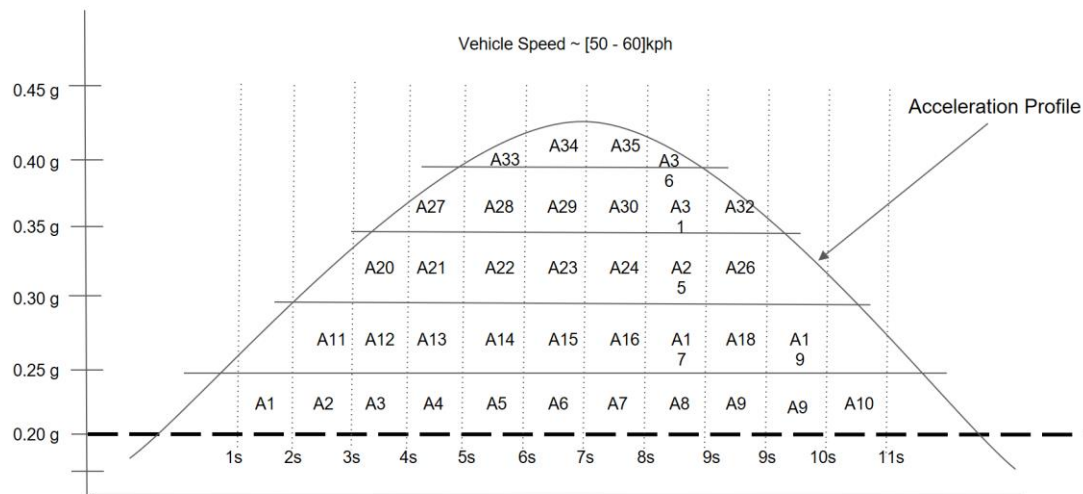


Figure 12: example for an acceleration profile

Penalty factors are assigned separately for each speed range as follows,

```
var accCarFactors = [[]float64{
    {0.00, 0.00, 0.00, 3.00, 5.00, 7.00, 9.00, 12.00}, /* 0-10 penalties */
    {0.00, 0.00, 0.00, 3.00, 5.00, 7.00, 9.00, 12.00}, /* 10-20 penalties */
    {0.00, 0.00, 3.00, 5.00, 7.00, 9.00, 12.00, 15.00}, /* 20-30 penalties */
    {0.00, 0.00, 3.00, 5.00, 7.00, 9.00, 12.00, 15.00}, /* 30-40 penalties */
    {0.00, 0.00, 5.00, 7.00, 9.00, 11.00, 13.00, 15.00}, /* 40-50 penalties */
    {0.00, 3.00, 5.00, 7.00, 9.00, 11.00, 13.00, 15.00}, /* 50-60 penalties */
    {0.00, 3.00, 5.00, 7.00, 9.00, 11.00, 13.00, 15.00}, /* 60-70 penalties */
    {0.00, 3.00, 5.00, 7.00, 9.00, 11.00, 13.00, 15.00}, /* 70-80 penalties */
    {0.00, 3.00, 5.00, 7.00, 9.00, 11.00, 13.00, 15.00}, /* 80-90 penalties */
    {3.00, 5.00, 7.00, 9.00, 11.00, 13.00, 15.00, 21.00}, /* 90-100 penalties */
    {3.00, 5.00, 7.00, 9.00, 11.00, 13.00, 15.00, 21.00}, /* 100-110 penalties */
    {3.00, 5.00, 7.00, 9.00, 11.00, 13.00, 15.00, 21.00}, /* 110-120 penalties */
    {3.00, 5.00, 7.00, 9.00, 11.00, 13.00, 15.00, 21.00}, /* 120-180 penalties */
}
```

Figure 13: Penalty factors



For each interval above threshold level, the penalty factor increases. For the example of Fig.12, for areas between A1 and A10 (for acceleration range between 0.2g-0.25g), the penalty factor is 3; for areas between A11 and A19 (acceleration range 0.25g-0.3g), 5, and so on.

The formula for the score and the related penalty factor calculation is shown below,

$$\text{SCORE} = 100 - (\text{Penalty Factor} / \text{total acceleration time}) * 100$$

$$\text{Penalty Factor} = (A1+A2+...+A10) * 3 + (A11+A12+...+A19) * 5 + (A20+A21+...+A26) * 7 + (A27+A28+...+A32) * 9 + (A33+A34+A35+A36) * 11$$

## 5.2. Recommendation system

At the end of each trip total and individual scores are calculated and presented to the driver in mainly five categories, total, acceleration, braking, idling and speeding behaviours. Then related messages are given in different languages in line with the test sites, below an example of the messages are given. For idling no messages are given however the score will be shown.

**Table 3: Recommendation system**

GOOD_ECO_DRIVING	You are doing pretty well, driving in a way that limits your emissions. Thank you!
ORANGE_ACCELERATION	You seem to accelerate quite frequently, which increases your emissions. Try to limit your acceleration, especially if you see that you will need to slow down very soon afterwards.
RED_ACCELERATION	You seem to accelerate harshly or aggressively, which seriously increases emissions. Try to limit your acceleration, and accelerate more gradually.
ORANGE_DECELERATION	You seem to brake quite hard, which increases emissions from your brakes and tyres. Try to anticipate more in advance situations where you need to slow down or stop, so that your braking is more gentle.
RED_DECELERATION	You seem to brake very harshly, which seriously increases emissions from your brakes and tyres. Try to keep a lower speed in order to anticipate in advance situations where you need to slow down or stop, so that your braking is more gentle.
ORANGE_SPEEDING	Your driving is quite fast on occasions, which increases your exhaust emissions. Limit your speed in order to reduce emissions and also save fuel.
RED_SPEEDING	Your driving is often at excessive speeds, which seriously increases your exhaust emissions. Limit your speed in order to reduce emissions and also save fuel, and keep within the legal speed limits.

Orange is triggered if the score is between 70-90  
Red is triggered if the score is less than 70

The screenshot in Figure 14 shows the display on the mobile application.

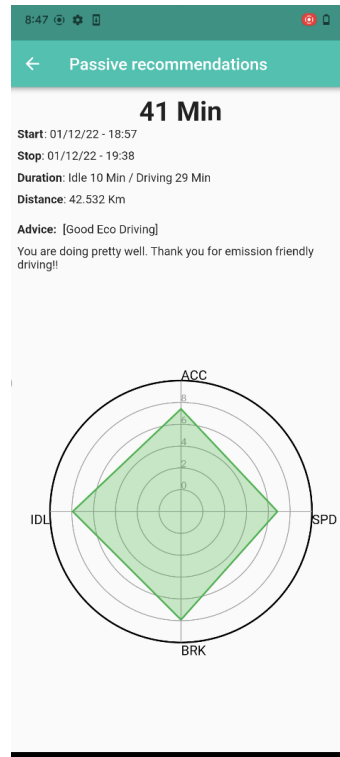


Figure 14: Passive recommendation system

## 6. The web dashboard

The mobile application is complemented by a dashboard web application mainly addressed local authorities and potentially for the public too, providing them with a statistical overview of the data collected.

For each app user, anonymous indicators are transmitted to a dashboard application to collect usage statistics and performance metrics. The latter allows the authorities and the public to understand the benefits of the mobile app, and view statistics by region or type of user. The dashboard application aggregates the data and present them in various graphical representations in order to assist the decision making.

The dashboard application is based on the data received from the app server and more precisely on some significant and indicative indicators. The indicators are:

- Vehicle emissions
- Fuel Consumption
- Driver's low emission driving performance

The dashboard web application's system architecture follows the N-tier architecture. The application exploits the web services for providing data from/to the database. The web services have been built according to the most popular architectural styles that is Representational State Transfer (REST) architecture. The implementation of the web services is based on Java programming language and more precisely on the Java Jersey framework. Additionally, the user interface (presentation layer) has been implemented based on the Grafana open-source analytics and visualisation software, the HTML, CSS and JavaScript programming languages. Grafana offers an open-source tool for building dashboards, query data sources, explore, monitor and visualise metrics.

The web application can be accessed by authorised users (Figure 15). There are two types of users. Each user may be either an ordinary user or a user belonging to an authority. User accounts are generated by the system administrator.

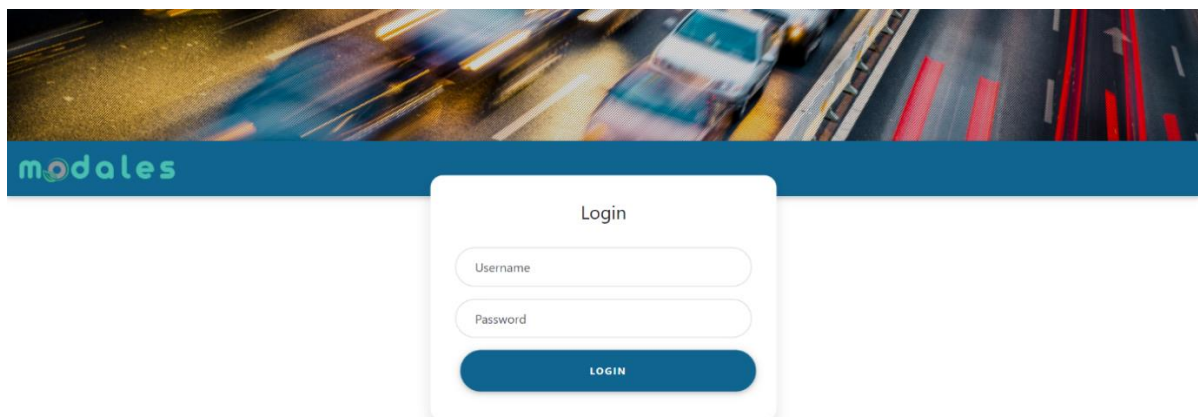


Figure 15: Dashboard's login page

The web application's user interface consists of three major sections. The first, called *dashboard* (Figure 16), presents the overall dashboard including aggregated indicators from all data collected. The aggregated indicators are related to the average fuel consumption and average NOx emissions per emissions label, per vehicle category, per road type, per license category, per driving experience, per gender and per age group.



Figure 16: Overall dashboard

The second section, called *region* (Figure 17), provides also aggregated indicators according to the data gathered from a specific region. Users belonging to a specific authority (responsible for a region) can only access the aggregated information for that region. The information presented is similar to the one in the overall dashboard page. Moreover, there is aggregated information about the total kilometres per road type and the average fuel consumption per road type.



Figure 17: Dashboard per Region

The third section, called *warnings* (Figure 18), provides valuable information about warnings. This web page provides aggregated information on how many vehicles are experiencing possible tampering and maintenance issues. The user accessing this page can view detailed information about the vehicle's brand and model and the description for the warning. The subsystem of the warnings is described in detail in MODALES deliverable *D4.2 Recommendations for anti-tampering and an improved mandatory vehicle inspection*.

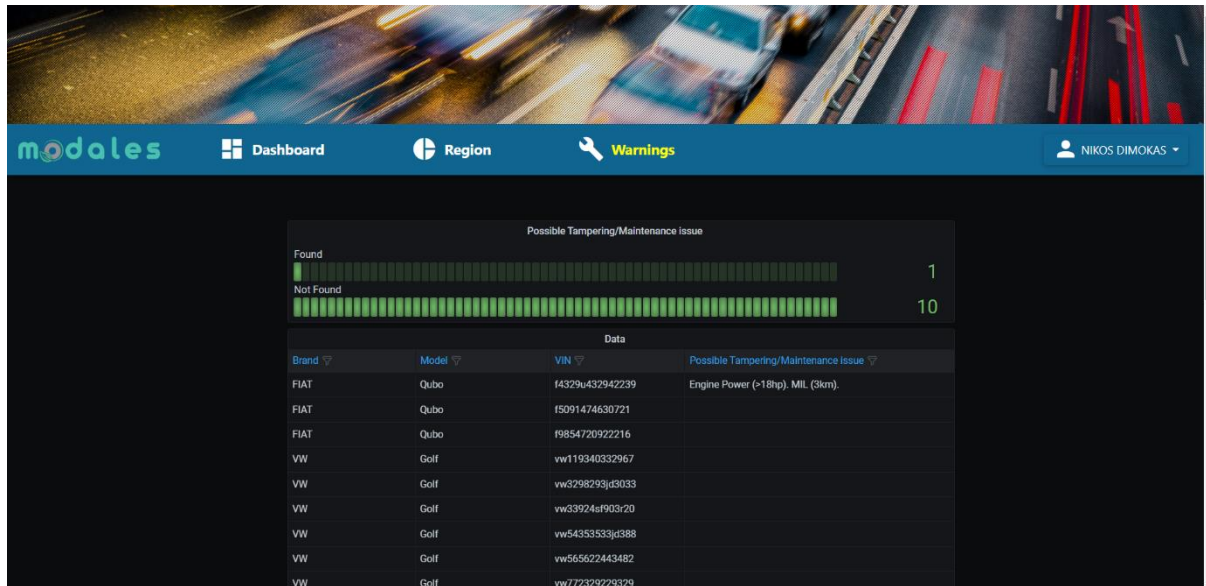


Figure 18: Warnings

The private user could potentially have access to the dashboard, but only to his/her information (Figure 19). More precisely, the user is able to see the total kilometres per road type travelled with his/her vehicle. Additionally, the web application presents information about the average fuel consumption and NOx emissions per road type.

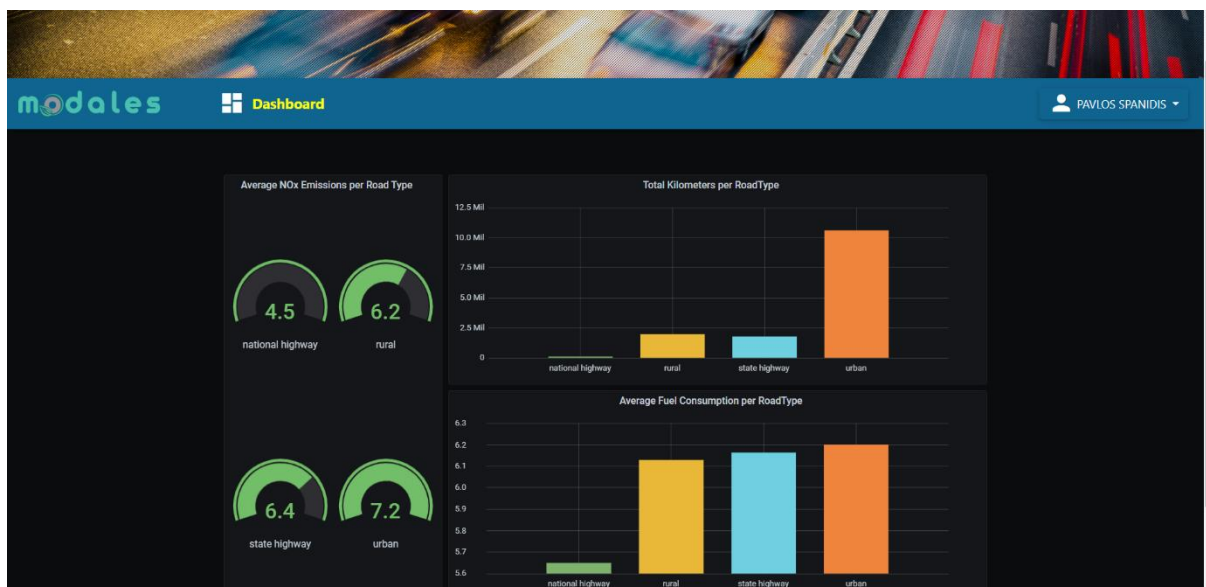


Figure 19: User's dashboard

To produce the aggregated data, the dashboard application also needs a data storage infrastructure. The MySQL Relational Database Management System has been used to store the data. The data model



was generated as a relational schema made up of multiple tables and the connections that exist between them. The tables and related properties (columns) are shown in the entity relational model below.

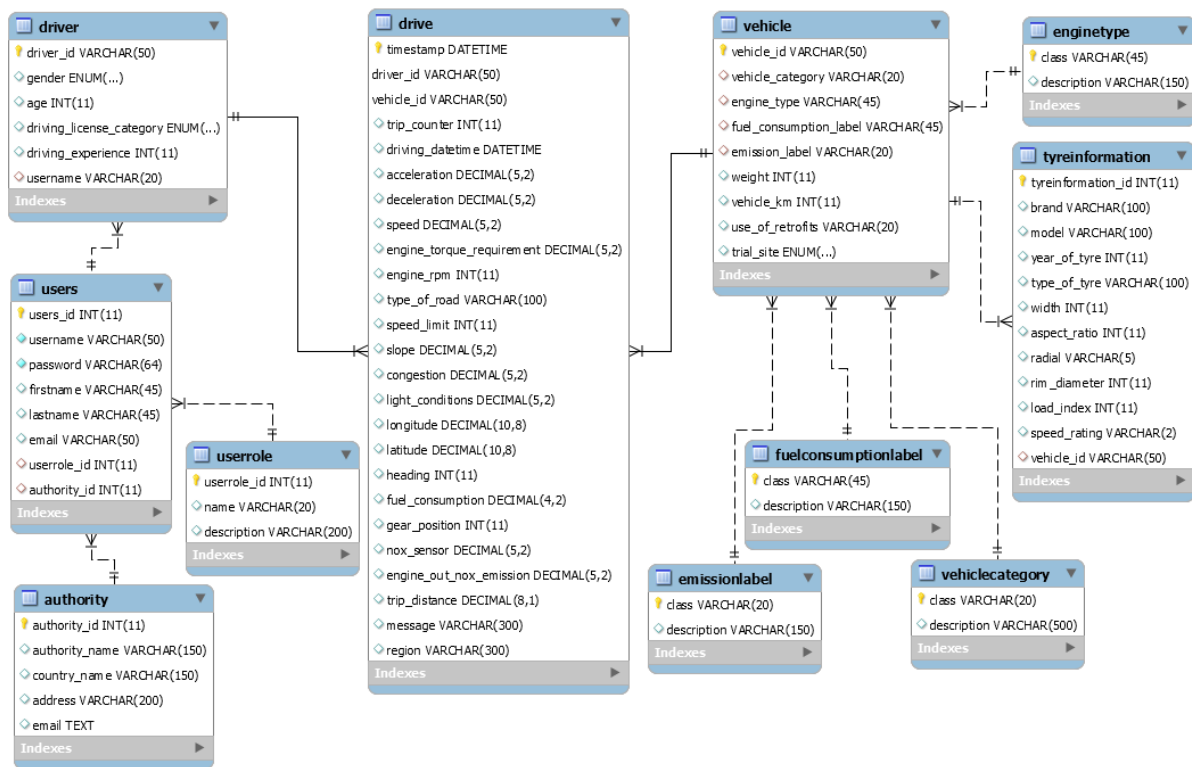


Figure 20: Entity Relational Model of dashboard's data storage

While the *users*, *userrole*, and *authority* tables save data about users, the *driver* table stores information about drivers. The dashboard web application needs the user details to implement the authentication process. The tables *enginetype*, *fuelconsumptionlabel*, *emissionlabel*, and *vehiclecategory* hold additional data, such as the various emission labels or the vehicles' categories, whereas the tables *vehicle* and *tyreinformation* store information specific to each vehicle. The *drive* table stores the information from each journey. Due to the fact that it retains all pertinent information of the driver while on the road, the drive table has a significant number of records. Based on how frequently the mobile application receives data from the OBD, data frequency is determined.

Finally, the integration of the web application with the rest of the system is realised with the implementation of a RESTful web service and a JSON data schema. The Application Program Interface (API) describing the integration and more precisely how the server sends the data to the dashboard web application is presented below. The data are formatted as a JSON text and contain information from the OBD, the vehicle and the driver. The server calls the API after the journey ends in order to provide all the data generated. After receiving the data, the application processes it to transform it into a suitable format to be loaded in the database. The database contains a number of stored procedures that process the data to produce the aggregated information that is presented to the user through the user interface (presentation layer).

Table 4: Dashboard API for receiving data from the server

Operation Description	The dashboard's web service module is responsible for receiving the data, process and store them in the dashboard's database. The server sends to the dashboard's Back-End API module a JSON text containing the information from a trip. The server receives an HTTP code (e.g. 200, 400, 404, etc) about the operation.			
URL	<a href="http://160.40.60.237:8080/modales2.ws/rest/server/addMeasurements">http://160.40.60.237:8080/modales2.ws/rest/server/addMeasurements</a>			
Input	Format	Name	Type	Comments
JSON (HTTP POST)		driver	Object	The information relevant to the driver. The object contains the next five fields.
		uuid	String	The driver's unique identification
		gender	String	The driver's gender
		age	Integer	The driver's age
		driving_license_category	String	The driver's license category
		driving_experience	Integer	The driver's experience
		vehicle	Object	The information relevant to the vehicle. The object contains the next 11 fields.
		uuid	String	The unique identification of the vehicle
		manufacturer	String	The vehicle's manufacturer
		model	String	The vehicle's model
		category	String	The vehicle's category
		engine_type	String	The vehicle's engine type
		fuel_consumption	Object	The vehicle's fuel consumption. The object contains information about urban, extra urban and combined fuel consumption
		emission_label	String	The vehicle's emission label
		weight	String	The vehicle's weight
		tyre_information	Object	The vehicle's tyre information. It contains information about the brand, the model and the year.
		km	Integer	The vehicle's km
		trial_site	String	The trial site where the vehicle was
		journey	Object	The object contains basic attributes of the journey and a big table with records derived from the journey.
		uuid	String	The unique identification of the journey
		start	String	The starting date and time of the journey
		end	String	The ending date and time of the journey
		distance	String	The distance travelled
		messages	String Array	The messages computed based on driving behaviour
		records	Array	An array of journey's records. Each record contains the fields below.
		date	String	The record's date
		accelerometer	Object	The record's accelerometer values
		activity_recognition	Object	The record's activity recognition values

Operation Description	The dashboard's web service module is responsible for receiving the data, process and store them in the dashboard's database. The server sends to the dashboard's Back-End API module a JSON text containing the information from a trip. The server receives an HTTP code (e.g. 200, 400, 404, etc) about the operation.			
		bluetooth	Object	The record's Bluetooth values
		gps	Object	The record's GPS values
		gyroscope	Object	The record's gyroscope values
		obd	Object	The record's OBD values
		wifi	Object	The record's Wi-Fi values
		context	String	The record's context value
		region	String	The record's region value



<b>Usage Example</b>	<p>Example: Dashboard API for receiving data from the server  <a href="http://160.40.60.237:8080/modales2.ws/rest/server/addMeasurements">http://160.40.60.237:8080/modales2.ws/rest/server/addMeasurements</a></p> <p>Input JSON formatted as:</p> <pre>{   "driver": {     "uuid": "97c09b21-2af9-4676-a425-13876796c6a0",     "id": "...",     "gender": "male",     "age": 36,     "driving_license_category": null,     "driving_experience": 1   },   "vehicle": {     "uuid": "de5c42bc-ab3f-4a04-b27f-a973841dadab",     "manufacturer": "",     "model": "",     "category": "M",     "engine_type": "Petrol",     "fuel_consumption": {       "urban": null,       "extra_urban": null,       "combined": null     },     "emission_label": "UNKNOWN",     "weight": null,     "tyre_information": {       "brand": "",       "model": "",       "year": 2022     },     "km": null,     "trial_site": "Thessaloniki"   },   "journey": {     "uuid": "e2e0b909-37b9-46fe-9447-99b61493c84a",     "start": "2022-08-16T17:52:02+00:00",     "end": "2022-08-16T17:53:40+00:00",     "distance": 123.45,     "messages": [       "Test Recommendation 1",       "Test Recommendation 2"     ],     "records": [       {         "date": "2022-08-16T17:52:02+00:00",         "accelerometer": {           "samples": 1,           "mean_x": 0.17288143932819366,           "mean_y": 0.0972084179520607,           "mean_z": 9.910174369812012         },         "activity_recognition": {           "samples": "1",           "best_type": "IN_VEHICLE",           "max_confidence": "100.0"         }       }     ]   } }</pre>
----------------------	---

Operation Description	<p>The dashboard's web service module is responsible for receiving the data, process and store them in the dashboard's database. The server sends to the dashboard's Back-End API module a JSON text containing the information from a trip. The server receives an HTTP code (e.g. 200, 400, 404, etc) about the operation.</p>
	<pre> }, "bluetooth": {   "new_devices": 0,   "old_devices": 0,   "lost_devices": 0 }, "gps": {   "samples": 0,   "mean_altitude": null,   "mean_speed": null }, "gyroscope": {   "samples": 0,   "mean_x": null,   "mean_y": null,   "mean_z": null }, "obd": {   "samples": 0,   "mean_engine_coolant_temperature": null,   "mean_engine_speed": null,   "mean_vehicle_speed": null,   "mean_absolute_barometric_pressure": null,   "mean_catalyst_temperature_bank1_sensor1": null,   "mean_catalyst_temperature_bank2_sensor1": null,   "mean_catalyst_temperature_bank1_sensor2": null,   "mean_catalyst_temperature_bank2_sensor2": null,   "mean_ambient_air_temperature": null,   "mean_absolute_throttle_position_b": null,   "mean_absolute_throttle_position_c": null,   "mean_accelerator_pedal_position_d": null,   "mean_accelerator_pedal_position_e": null,   "mean_accelerator_pedal_position_f": null,   "mean_engine_fuel_rate": null,   "mean_maf_air_flow_rate": null }, "wifi": {   "new_devices": 0,   "old_devices": 0,   "lost_devices": 0 }, "context": null, "region": "Thessaloniki" } ] } } </pre>

## 7. Instructions for end-users: how to use the mobile app?

Instructions to use the app and launch a data collection are the following:

1. **Phone setup:** Phones should, if possible, be fixed in the car using a car phone holder so that you can have a comfortable view of the phone's interface, as well as for the proper recording of data.
2. **Downloading and installing:**
  1. **You have an Android phone:** the application is available at the following: [https://play.google.com/store/apps/details?id=lu.list.modales\\_app](https://play.google.com/store/apps/details?id=lu.list.modales_app) - please download the app and install it. The device must be an Android 11 (API level 30) or greater.
  2. **You have an iOS phone:** the application is available at the following: <https://testflight.apple.com/join/eTFPxyPk> - Please follow the instructions on the website. You will need to download TestFlight (the official beta-testing application of Apple), before installing the MODALES app. The device must be compatible with iOS 9 or greater.
3. **Opening the application for the first time:** When opened the first time, the application will require authorisations; please approve them. Afterward, you will have to fill in a form with your information. Please insert the code you received during the training in the "User ID" field.
4. **Button bar:** The main screen of the application shows a button bar at the bottom to access the "Journey" section (🚗) and the "Settings" section (⚙️).
5. **Journey:** In the Journey section, press the green play button to start the data collection. A red stop button will replace the green play button. Press the red stop button to end the data collection. The journey section also shows the OBD Dongle and Vehicle status bars. Press the gear button on each status bar to configure the corresponding features. Please note that some of the settings might be blocked during a journey.
6. **Configuring the OBD Dongle (optional):** If you have an OBD Dongle, you need to configure it first. Before starting, make sure that your phone has Bluetooth turned on. To connect to the OBD Dongle for the first time, you need to "pair" the dongle to the phone. You can do the pairing on the phone's Bluetooth menu or inside our application in the OBD Dongle setting screen. The OBD Dongle needs to be in discovery mode (e.g., on the OBDLink MX+, you can find a small button, keep it pressed until the blue light starts to blink). In the application, you can press the "Scan" button to search for the OBD Dongle, and when it appears on the screen, press the ( + ) to do the pairing. After pairing, you can press the ( ▶ ) to connect to the OBD Dongle for the first time. The application will attempt a connection to the last used OBD Dongle every time you start a journey (if Bluetooth is enabled). We recommend starting a journey after the vehicle ignition is on to avoid errors.
7. **Configuring the vehicle (optional):** The "Garage" section shows all the vehicles you have registered in the application. You can edit (✎), select (▶) and delete (🗑️) each vehicle, or create new vehicles (+). The application always needs one vehicle selected, and you cannot delete the selected vehicle. The application begins with a selected vehicle called "default vehicle" with empty details that you can complete. While creating or editing a vehicle, the application shows a form with all the vehicle details. When you connect an OBD Dongle, the application will try to obtain the vehicle identification number (VIN), and if possible, will try to select a vehicle containing the VIN or will create a new vehicle including the VIN.

8. **Configuring other parameters (optional):** If you want to change other aspects of the application, please check the "Settings" section (⚙️).
  1. **Server Synchronisation:** You can select how the application sends data from your phone to our servers. The default is "Only with Wi-Fi" because it does not require any action from you and because no paid service is required. As soon as the phone is connected to a WiFi network, an internal timer will periodically check (by default every 20 minutes) if there is data accumulated in the device that needs to be sent to our servers. The local data is periodically divided into smaller pieces to facilitate the transfer (in chunks of 1 MB every 1 minute by default). After a successful synchronisation, the data is deleted from the device. If the application is closed this synchronisation never takes place. If you chose "With Wi-Fi or Mobile Data", the behaviour is similar, but the mobile data is used when there is no Wi-Fi network available (this could be charge by your service provider depending on your plan). Finally, if you choose "Only Manually", a "Send" button will appear next to this option, and the synchronisation will only take place when you press the button.
  2. **Data Collection:** You can choose here to enable or disable any of the application sensors or data sources. By default, all of them are enabled to offer the best service possible.
  3. **Clear Stored Data:** You can press this option if you want to clean your local data (e.g., if the application is not synchronising for some reason and is taking too much space).
9. **Reporting an issue (optional):** In case of problems, please [contact](#) the project team.

## 8. Privacy and GDPR management

During the whole development cycle of the application, the data management was supervised by the project's data manager as well as by the LIST's data protection officer, who was responsible for the hosting of the data. Users of the mobile application are informed of the following:

### 8.1. Identity of the joint controllers

Joint controllers are the entities who jointly determine the purposes and means of processing of your personal data.

This project is coordinated by ERTICO – ITS Europe (European Road Transport Telematics Implementation Coordination Organisation), Avenue Louise 326, B-1050 Brussels, Belgium - <https://ertico.com>. The full list of partners in MODALES can be seen here: <https://modales-project.eu/about>

In relation to the personal data collected and/or processed via the present app, the Luxembourg Institute of Science and Technology (LIST) will act as main point of contact for inquiries by data subjects. LIST's offices are located at 5, avenue des Hauts Fourneaux in L-4362 Esch-sur-Alzette, Luxembourg.

The Data Protection Officer of LIST can be contacted via email: [dpo@list.lu](mailto:dpo@list.lu).

### 8.2. Categories of personal data we collect and purposes of processing

Below is a list of personal data collected:

- **Contact details** (your e-mail address): this information will be used in order to send you the link to download the app for the first time and to provide you with new versions of the app or automatic updates. Your e-mail will be stored separately from the dataset containing the remaining data and is only accessible to the partner responsible for your trial site.
- **Manual inputs from the user**: information about his/her vehicles, participant code, gender (optional), birthdate (optional), licence date (optional).
- Raw data from the smartphone's **accelerometer and gyroscope**. This data will be used to compute acceleration profile.
- **Activity data**, automatically computed locally on the phone to detect if the user is in a vehicle or not.
- **Anonymised Wi-Fi and Bluetooth data** collected **passively** from the other devices. This will be used for research purpose in order to detect the presence of a traffic jam or situational variables.
- **GPS positioning data**, which includes latitude, longitude and speed of the user collected at a regular sampling rate during the data collection. This GPS data will be sent to a server hosted by LIST in Luxembourg. It will be decoded and translated into anonymised indicators, including but not limited to: weather (e.g., temperature, humidity), road traffic (e.g., average traffic speed) and characteristics (e.g., slope, type of road) at each of the collected GPS data point. Only these decoded and anonymised indicators will be stored. All the GPS positioning data will be removed. An external service, managed by Motion-S in Luxembourg, will be used for this decoding. Motion-S is not storing any information - about users and their locations.

- **Vehicle information**, which will be retrieved through the Vehicle Identification Number (VIN) of the car used by the participant. This VIN will be collected automatically (if an OBD dongle is used) or manually (user input). The process and the conditions will be the same than for the GPS data. An external web service, [vindecoder.eu](http://vindecoder.eu), will be used to translate the VIN into useful and anonymised indicators. These indicators include, but are not limited to: make of the car, manufacturer, type of engine, vehicle type, etc.
- **OBD data**, collected through the OBD dongle. Various variables are collected, and heavily depend on the type of vehicles. Most common variables include, for instance: revolution per minute, gas pedal position, fuel consumption, vehicle speed, NOx emission, gear position at a given time.

## 9. Conclusions

This deliverable is submitted before the end of the data collection campaign (WP6). Some development elements and conclusions will therefore have to be analysed with deliverable D5.3, once the campaigns are fully completed. In parallel, it is important to note that deliverable D5.4 is very complementary to this deliverable and includes some testing elements that we encourage interested readers to go through.

The development of this mobile application was an interesting exercise in the context of the project, which addressed the following challenges:

- **Technical challenges:**

- No open source OBD library for Flutter
- The OBD dongle only works with legacy Bluetooth (serial)
- Differences between emulator and vehicles
- Differences between manufacturers
- Differences between OBD protocols (around 10)
- The phone has no fixed position
- Different sample ratios between sensors
- Some sensors are optional
- Unexpected differences between Android and iOS

- **Research challenges:**

- Learn useful driving indicators from raw data in real time.
- Design a user interface for the journey without compromising safety.
- Build a recommender system.
- Handle different connectivity scenarios.
- Ensure privacy policies.
- Analyse behaviour changes.

**For more information:**

MODALES Project Coordinator:

ERTICO - ITS Europe

Avenue Louise 326

1050 Brussels

[info@modales-project.eu](mailto:info@modales-project.eu)

[www.modales-project.eu](http://www.modales-project.eu)



**Adapting driver behaviour  
for lower emissions**



This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 815189.