# PDP
# DOCUMENTATION INDEX

# SUPPORT

Need further help?

Drop an email to one of the following:

**Collaborations & big decisions:**
*Boss Man :*
boss.man@oem.com

**Technical:**
*Architect*
architect@oem.com
*Development lead*
dev.lead@oem.com

**Product features, wishlists, clarifications:**
*Product boy*
Product.boy@oem.com

*P.S: For providing your feedback, pl scroll to the bottom of any module's page or click here. We are all ears.*

# API INDEX BY TYPE

### These are the QUERY APIs:



### These are the MUTATION APIs



### These are the SUBSCRIPTION APIs

# INTRO TO PDP

Our warmest welcome to you to the developers' portal for PDP!

Let's quickly get your engine revving or motor humming (whichever side you choose :-P )

**Did you know?**
Each vehicle generates a row of data each second it is ON…
Each row of data is 100+ fields…

PDP helps you unlock the immense potential of this gold mine!
We help make sense of big data, streaming continuously while the vehicles drive around & even when they simply stand.

PDP is a B2C, data exchange platform that serves as a robust, easy-to-use, decoupled backend for any data that is generated by vehicles.

With our ever-growing collection of APIs, upgraded to the industry's new favorite GraphQL, you can rest assured that the heavy lifting is done & all that's left to do is to integrate the APIs seamlessly & deliver the best functionalities to your users.

Be it fully-functional mobile apps or browser-based portals or Whatsapp integrations or integrations with wearables and smart devices like Alexa, PDP shall help you get up & running in the fastest way.

# How PDP differs from After-market devices?

Good & valid question.

PDP brings with it OEM's understanding of vehicle data gathered over countless, relentless years of passionate product development.

On PDP, you can onboard 20+ after market devices readily, out-of-the-box. It's device agnostic. Want a GoMechanic device on your Apache and OneLap on your Pulsar? Very much possible…

PDP lets you have one view for your multiple vehicles, how many ever wheels too!
It is product agnostic. Fancy a garage? PDP is your garage for your data & integrations.

You can even choose to have your own protocol / payload structure and your device can be onboarded onto the platform. It's protocol agnostic.

And that's why we say, its generic yet specific, how you see it.

PDP APIs are made to support MACH. They are modular and can support several front-ends & several products with micro-services and a backend decoupled from frontend.

---

# PDP’s APIs RePEAT

We built our APIs around what we call RePEAT framework.

**Re**liable
**P**rocessed
**E**nriched
**A**ggregated
**T**imebound

The APIs are designed such that they are product-agnostic yet at the same time, serve even specific special requirements of each product.

The APIs have been divided into modules based on the umbrella of use-cases that they serve. So, if you had any use-cases with real-time data from vehicle, all the APIs serving those will be grouped together.

# Excited…? So are we, to get you started.

Getting started

## Quick brief about GraphQL

In simple terms, GraphQL APIs work with a single end-point as compared to REST which have multiple endpoints & URIs based on the desired outcome. This allows for combining information fetches from database instead of multiple REST calls. GraphQL also works like a query language, and unlike REST, in GraphQL, you can choose which fields to receive. Thus undersharing and oversharing are addressed effectively. REST have GET,POST,PUT, PATCH, DELETE as methods. GraphQL has Query, Mutation and Subscription.
Both use HTTPS to send request and receive response.
By default, GraphQL, when RESTified, work with POST method & we understand if you felt this weird.

## GraphQL Methods

**QUERY** shall be used for getting any information,
**MUTATION** shall be used for posting, putting, patching and deleting,
**SUBSCRIPTION** is used to receive live, streaming data without repeatedly calling a GET API (polling).

You won't have to call different endpoints now to get info about a user's mobile number, a user's vehicle details and a user's subscription status, for example. This operation would have normally taken 3 separate API calls with REST. The major difference that can be felt is in the way of working with GraphQL. Unlike REST, GraphQL APIs cannot be called from browser windows & clients directly and they don't conform to OpenAPI specification. However, there are specific clients like Altair Client, GraphiQL, GraphQL Playground to make this easy. The links help you download a Chrome extension for each of these.

You can also make GraphQL requests with Postman. GraphQL query goes into the body portion & the remaining works like a charm. You'll find plenty of examples starting from Authorization API to understand how to make these calls.

For a more detailed description of GraphQL vs REST APIs, we recommend this read:
GraphQL vs REST - A comparison

If you fancy directly getting your hands dirty with the APIs, try running the examples provided for each API on your local (Python/JS) or on a client extention like Altair Client, GraphiQL, GraphQL Playground.

# Modules

We've divided our API collection into modules based on their functions.

A 🔍Glossary of modules & APIs introduces each module and its overall function.

## Brief description of modules of PDP

| Sl.no. | Module | Description |
|---|---|---|
| 1 | Authentication | This module provides details on how to generate a bearer token which shall be required as Header in all API calls. |
| 2 | User, vehicle onboarding & user, vehicle details | This module consists of APIs that help onboard a new vehicle and new users and manage them on PDP |
| 3 | Live tracking | This module consists of APIs that share live data of parameters most useful to the end customer. |
| 4 | Favorites | This module consists of APIs that help manage a user's favorite locations for navigation use-cases. |
| 5 | Recent navigations | This module consists of APIs that let a location be tagged as a recent navigation/search |
| 6 | Share location to vehicle | This module consists of APIs that let a location be shared to a vehicle from any front-end |
| 7 | Geofence | |
| 8 | Trips summaries | |
| 9 | Charging summaries | |
| 10 | Driver scoring | |
| 12 | Charging stations | This module consists of APIs that share live details about charging stations, a specific use-case for EV products. |
| 13 | Crash/fall alerts | |
| 14 | Theft/tow/tamper alerts | |
| 15 | Conditional maintenance alerts | |
| 19 | Profile settings sync | |
| 20 | Vehicle health report | |
| 21 | Parental controls | |
| 22 | Incognito mode | |
| 24 | Data explorer | |