

**Juganu Data API**

March 2025

Table of Contents

[Introduction 2](#_Toc194484612)

[Terminology 3](#_Toc194484613)

[Marking Configuration Tool 4](#_Toc194484614)

[API Lists 5](#_Toc194484615)

# 

# Introduction

Juganu’s platform enables advanced AI-powered analytics by transforming physical spaces into digitally monitored environments using edge computing, computer vision, and AI. Our API provides access to the data, allowing developers to build custom applications, analytics, and automation layers on top of our infrastructure.

This section introduces the key concepts and terminology necessary to understand how different AI analytics DATA metrics are structured within the API. By leveraging this knowledge, developers can efficiently integrate Juganu’s data streams, event-based notifications, and AI-powered insights into their solutions.

If you identify any missing use cases that cannot be addressed using the APIs described in this document, please feel free to contact us. We’ll work with you to find the optimal way to retrieve the required data, and if necessary, creating a new API.

# 

# Terminology

* **Site**   
  A site in Juganu’s system refers to an independent location, such as a region, parking area, gas station, or city zone. Each site operates independently, collecting and processing data locally
* **Object Of Interest (OOI)**  
  A tracked entity in the system, such as a person, car, truck, or other object which was trained to be detected by our edge processor.
* **Projection**Projection is a feature that maps detected objects from the camera frame to a bird’s-eye view representation. This process converts the object’s position from its original (x1, y1) coordinates in the camera frame to a (x2, y2) location on a reference surface.
  + Outdoor Projection → The bird’s-eye view is based on a latitude/longitude (lat, lng) of Google map.
  + Indoor Projection → The bird’s-eye view is based on an x, y floor plan.

This transformation enables spatial tracking of objects across different perspectives, tracking objects over time, and generate meaningful spatial insights

* **Visit**

A visit is defined as the appearance of an object in a camera or a stations for a minimum duration (e.g., 10 seconds). This threshold is primarily used to filter out false detections and is configurable.

* **Metrics**The Metrics table stores aggregated statistics derived from visits, grouped into time buckets (minutely, hourly, daily). These metrics provide insights such as peak activity times in different stations, cameras or for a site.
* Metrics Examples:
  + Total OOI count per a given time range – The total number of Objects of Interest (OOIs) detected within a given period.
  + Peak and minimum OOI count within a single frame – The highest and lowest number of objects present at the same moment during a specified timeframe.
  + Data aggregation – Metrics can be analyzed by OOI type, station, camera or a full site to provide relevant insights.
* Examples of Use Cases:
  + Example 1: Intersection Peak Load Analysis
    - Goal: Identify the busiest moments at an intersection to optimize traffic management.
      * Metric Used: Peak OOI count within a time period – Determines the peak vehicle load at a specific moment.
      * Turn it to your insight: If an intersection reaches its highest congestion with 50 vehicles at 5:30 PM, this data can be used to adjust traffic light timing and improve flow efficiency.

# Marking Configuration Tool

Juganu has developed a marking tool, designed as a configuration utility for defining areas and lines within the scene. In addition to marking zones of interest, the tool allows users to define exclusion zones (Filter ROIs) areas that should be ignored by the edge device during object detection.  
The marked areas and lines are later utilized to retrieve important statistical insights and/or near real-time metrics.

* **Station**  
  A **polygon** defined on a camera view or map, used for **logical grouping** of detections.
  + Example use case: Marking **restricted areas**
* **Line**A **line drawn on a camera view** to count **how many OOIs cross it.**Example use case: Counting vehicles crossing an **intersection from one side to the other.**

### 

# API Lists

This document comes with a package that allows you to open it in a web view and see the list of available APIs.  
You’ll need to download the ZIP file attached to this document and open the HTML file included inside.

API Usage Examples

**Login (REST)**

Endpoint: POST <https://prod.api.smart.juganu.com/api/v1/login>

Request Body:

{"email":"example\_username@juganu.com","password":"example\_password"}

A screenshot of a computer

Description automatically generated

**Get a list of stations**

Endpoint: <https://prod.api.smart.juganu.com/api/v1/graphql>

Request Body:

**query** get\_stations {

    get\_stations {

        id

        name

        is\_activated

        is\_bird\_view\_mode

    }

}

**A screenshot of a computer

Description automatically generated**

**Get lines**

Endpoint: <https://prod.api.smart.juganu.com/api/v1/graphql>

Request Body:

**query** lines {

    get\_lines {

        id

        name

        is\_activated

        detector\_type

        directions {

            id

**type**

        }

    }

}

A screenshot of a computer

Description automatically generated

**Get cameras (Including snapshot and stream)**

Endpoint: <https://prod.api.smart.juganu.com/api/v1/graphql>

Request Body:

**query** Get\_cameras {

    get\_cameras {

        id

        name

**type**

        mac\_address

        device\_mac\_address

        device\_name

        stream\_urls {

            address

**type**

        }

        snapshot\_image {

            url

        }

    }

}

A screenshot of a computer

Description automatically generated

**List of Visits (cars) in a specific station based on name during specific date/time**

Endpoint: <https://prod.api.smart.juganu.com/api/v1/graphql>

Request Body:

**query** Query\_visits {

    query\_visits(

        filter\_by: {

            and: [

                {

                    between: {

                        field: "start\_datetime"

                        value: { start: "2025-03-28T12:00:00", end: "2025-03-28T13:00:00" }

                    }

                }

                { equal: { field: "detector\_type", value: "car" } }

                { equal: { field: "station\_visits.station.name", value: "Gravel" } }

            ]

        }

    ) {

        start\_datetime

        end\_datetime

        duration\_seconds

        detector\_type

        station\_visits {

            start\_datetime

            end\_datetime

            duration\_seconds

            station {

                id

                name

                is\_activated

            }

        }

    }

}

**A screenshot of a computer

Description automatically generated**

**Get Aggregated (Minutely) list of People (grouped by camera, station & site)**

Endpoint: <https://prod.api.smart.juganu.com/api/v1/graphql>

Request Body:

query Query\_metrics {

query\_metrics(

filter\_by: {

and: [

{

between: {

field: "start\_datetime"

value: { start: "2025-03-28T09:18:12Z", end: "2025-03-28T10:18:12Z" }

}

},

{

equal: {

field: "ooi\_type"

value: "body"

}

}

]

}

time\_bucket: Minutely

) {

start\_datetime

end\_datetime

ooi\_type

group\_type

group\_id

occupied\_percent

unique\_ooi\_count

ooi\_count

max\_duration

avg\_duration

}

}

A screenshot of a computer

Description automatically generated

**Get Aggregated (Hourly) list of Cars grouped by site**

Endpoint: <https://prod.api.smart.juganu.com/api/v1/graphql>

Request Body:

**query** Query\_metrics {

  query\_metrics(

    filter\_by: {

      and: [

        {

          between: {

            field: "start\_datetime"

            value: { start: "2025-03-28T09:00:00", end: "2025-03-28T15:00:00" }

          }

        },

        {

          equal: {

            field: "ooi\_type"

            value: "car"

          }

        },

        {

          equal: {

            field: "group\_type"

            value: "site"

          }

        }

      ]

    }

    time\_bucket: Hourly

  ) {

    start\_datetime

    end\_datetime

    ooi\_type

    group\_type

    group\_id

    occupied\_percent

    unique\_ooi\_count

    ooi\_count

    max\_duration

    avg\_duration

  }

}

A screenshot of a computer

Description automatically generated

**Get List of cars between dates that passed in a specific line by id**

Endpoint: <https://prod.api.smart.juganu.com/api/v1/graphql>

Request Body:

query Query\_visits {

query\_visits(

filter\_by: {

and: [

{

between: {

field: "start\_datetime"

value: { start: "2025-03-20", end: "2025-03-30" }

}

}

{ equal: { field: "detector\_type", value: "car" } }

{ equal: { field: "line\_visits.line.id", value: "EtVCtY7mfmFqPJynmbzz" } }

]

}

) {

start\_datetime

end\_datetime

duration\_seconds

detector\_type

line\_visits {

line {

id

name

}

}

}

}

A screenshot of a computer

Description automatically generated

**Get notifications (REST)**

Endpoint: [https://prod.api.smart.juganu.com/api/v1/notifications/notifications](https://staging.api.smart.juganu.com/api/v1/notifications/notifications)

Query params:

Time - time in epoch of since when to get notifications

Limit - limit the number of notifications in the response

Sort - order the notifications

Example:

GET <https://prod.api.smart.juganu.com/api/v1/notifications/notifications?time=1732715819&limit=10&sort=timestamp:desc>

**Create Filter ROI (REST)**

Endpoint: POST [https://prod.api.smart.juganu.com/api/v1/configuration/roi/filter\_roi](https://staging.api.smart.juganu.com/api/v1/configuration/roi/filter_roi)

Request Body:

{

"id": null,

"name": "FilterROI1",

"camera\_id": "u6iETwrgACsmkyfpPJVU",

"camera\_location": [

[

139.3170731707317,

616.0239248253431

],

[

315.5121951219512,

886.3535836580154

],

[

139.3170731707317,

882.2576797363083

]

],

"is\_activated": true

}

**Get Filter ROI (REST)**

Endpoint: GET <https://prod.api.smart.juganu.com/api/v1/configuration/roi/filter_roi>

A screenshot of a computer

Description automatically generated

**Create Station on Site view**

**mutation** Create\_stations {

create\_stations(

stations: {

name: "StationName"

logical\_groups: ["11111111111111111111"]

is\_activated: **true**

is\_bird\_view\_mode: **true**

camera\_view\_locations: []

bird\_view\_location: {

**type**: "Feature"

properties: {}

geometry: {

**type**: "Polygon"

coordinates: [

[

[34.85967211992139, 32.020390030692006]

[34.85968751556055, 32.02034303866206]

[34.85974601898949, 32.02037958802076]

[34.85970906945542, 32.02041874803177]

[34.859669040793555, 32.02040830536381]

]

]

}

}

}

) {

id

name

is\_activated

is\_bird\_view\_mode

bird\_view\_location

camera\_view\_locations {

location

camera {

id

name

device\_name

device\_mac\_address

}

}

logical\_groups {

id

**type**

}

image {

url

}

}

}

**Get a list of visits of a specific OOI (of type: car)**

Endpoint: <https://prod.api.smart.juganu.com/api/v1/graphql>

Request Body:

**query** Query\_visits {

query\_visits(

filter\_by: {

and: [

{

between: {

field: "start\_datetime"

value: { start: "2025-03-28", end: "2025-03-29" }

}

}

{ equal: { field: "detector\_type", value: "car" } }

{ equal: { field: "ooi.id", value: "144ea812-8804-4d11-898a-21883e9f6789" } }

]

}

) {

start\_datetime

end\_datetime

duration\_seconds

site\_id

site\_name

customer\_id

detector\_type

cameras {

id

name

**type**

mac\_address

device\_mac\_address

device\_name

}

ooi {

id

name

**type**

}

}

}

A screenshot of a computer

Description automatically generated