OpSense Mobile v2 API Reference

Complete endpoint reference for the OpSense mobile app (v2)

opsenseapi · Version 1.0 · 6 November 2026

1. Introduction

This document lists every service the OpSense mobile app (v2) calls: what it is for, every parameter, every response field and every error. Use it together with the change guide, which only lists what is new in v2.

1.1 Requests

Every service is one call:

```
POST https://<api-host>/v1/<service>
Content-Type: application/x-www-form-urlencoded
```

The service name is the last part of the URL, for example `SRVGetWorkOrders`. Send every parameter as a **form field**, including `api_key`. A JSON request body is not read: parameters sent as JSON are ignored, and the service answers as if they were not sent. Files (videos, images) are sent as `multipart/form-data`. Parameters in the URL query string are only read when the request has no form fields at all, so do not rely on them.

Parameters that are not listed for a service are ignored.

1.2 Responses and errors

Every response is wrapped the same way:

```
{"result": { ... success or error body ... }}
```

A success body has `"result": "success"` (read services) or `"result": "1"` (services that save or delete). An error body has `error_code` and `error_text`. `error_code` is a string, for example `"10018"`.

An unknown service name returns `{"error": "No routing. ..."}` without the `result` wrapper.

These errors can come from every service:

`error_code`	Meaning
10015	`api_key` missing.
10016	`api_key` invalid or inactive.
10099	The request could not be completed on our side. The body is `{"error_code": "10099", "error_text": "Request could not be completed. Please try again."}`. Safe to retry a read. For a save, load the data first to check whether it was saved.

Each service below lists its own codes. Several services use the same number for different things (10019, 10020), so always read the code together with the service.

1.3 Authentication

Send the company's `api_key` with every call. The key decides the company. You only ever read or change that company's data.

1.4 Dates and time zones

The company has one time zone (see `SRVGetTimezone`). The database keeps all times in server time (Europe/London). The services convert as follows:

- Dates you send to a field that says "company time zone" are converted to server time before they are saved.
- Dates in a response are converted from server time to the company time zone, but only for the fields the service lists as converted. Fields not listed as converted come back as stored, in server time.
- Format is always `YYYY-MM-DD HH:MM:SS` for date-times and `YYYY-MM-DD` for dates.

Where a service does not convert a date, its section says so. For a company whose time zone is Europe/London there is no difference.

1.5 Numbers and ids

Ids and numbers may arrive as strings (`"3492"`) or as numbers (`3492`), depending on the field. Always parse them. Fields a service builds itself (for example `avg_minutes`, `has_more`) are real numbers and booleans.

An empty list is `[]`. A missing value is `null` or an empty string.

2. Screen map

Mobile v2 screen	Endpoints
Home project cards and Project Details	`SRVGetProperties`, `SRVGetWorkOrders`
Work order detail	`SRVGetWorkOrders` with `reference_id`, `SRVGetWorkOrderNotes`, `SRVAddWorkOrderNote`, `SRVUpdateWorkOrderNote`, `SRVDeleteWorkOrderNote`, `SRVDeleteWorkOrder`, `SRVAddWorkOrder` (status and videos)
Work order list	`SRVGetWorkOrders` with `limit` and `before_id`
New / edit work order form	`SRVAddWorkOrder`, `SRVGetCustomers`, `SRVGetProperties`, `SRVGetServices`, `SRVGetProducts`, `SRVGetStockLocations`, `SRVGetRoomTypes`, `SRVGetTimezone`; new customer: `SRVAddCustomer`, `SRVAddProperty`, `SRVAddRoomFlat`
Calendar	`SRVGetWorkOrders` with `appoinment_start_date` and `appoinment_end_date`, `SRVGetWeeklyRoute`
Notifications	`SRVGetNotifications`, `SRVMarkNotificationsRead`

Operations dashboard	`SRVGetServiceDurations` (task durations), `SRVGetNotifications` (panic alerts are notifications whose `title` starts with `PANIC BUTTON - `), `SRVGetWeeklyRoute`, `SRVGetWorkOrders`
Staff status	`SRVGetWeeklyRoute`
Module menu and permissions	`SRVGetUserRoles`
Panic button	`SRVPanicButton`
AI Visor report	`SRVGetAiVisorOverview`, `SRVGetAiVisorReport`

Beacon, reader, live staff location, device approval and AI chat screens are served by the Mopuan backend and are not part of this API.

3. Work orders

3.1 SRVGetWorkOrders

Returns the company's work orders with their customer, service, building, room, assignee, products and notes. Use `reference_id` for the detail screen, `limit` and `before_id` for the list, and the appointment range for the calendar.

Field	Required	Notes
`api_key`	**Required**	See section 1.3.
`reference_id`	Optional	Return only this work order.
`appoinment_start_date`	Optional	Appointments from this date and time. Note the spelling: `appoinment`. Not converted: the value is compared as sent with the stored appointment (server time).
`appoinment_end_date`	Optional	Appointments up to this date and time. Same rule as the start date.
`room_id`	Optional	Only this room's work orders.
`mopuan_employee_id`	Optional	Only work orders assigned to the user with this Mopuan id.
`limit`	Optional	Page size. Returns the newest `limit` work orders (highest id first) and adds `has_more` and `next_before_id`.
`before_id`	Optional	Only work orders with an id lower than this. Send the `next_before_id` of the previous page.

Without `limit` and `before_id` every matching work order is returned, in no guaranteed order, with no paging fields. With `before_id` and no `limit` the older work orders are returned newest first, with no paging fields.

Paging: ask for the first page with `limit` and no `before_id`. While `has_more` is `true`, ask for the next page with the same filters, the same `limit` and `before_id` set to `next_before_id`. To refresh, ask for the first page again.

```
curl -X POST "https://<api-host>/v1/SRVGetWorkOrders" \
 -d "api_key=<api_key>" \
 -d "limit=20"
```

```
curl -X POST "https://<api-host>/v1/SRVGetWorkOrders" \
 -d "api_key=<api_key>" \
 -d "appoinment_start_date=2026-11-01 00:00:00" \
 -d "appoinment_end_date=2026-11-30 23:59:59"
```

Response (one work order shown):


```
{
 "result": {
   "result": "success",
   "data": [
     {
       "id": "3492",
       "company_id": "7",
       "customer_id": "55",
       "branch": "12",
       "room": "301",
       "service": "4",
       "assigned_to": "88",
       "assigned_by": "3",
       "assigned_date": null,
       "appoinment_date": "2026-11-06 09:00:00",
       "added_date": "2026-11-05 16:40:12",
       "added_by": "3",
       "status": "3",
       "video_before": "work-orders/7/2026/11/3492_before_6540a1.mp4",
       "video": "work-orders/7/2026/11/3492_proof_6540b2.mp4",
       "priority": "1",
       "is_vip": "0",
       "start_date": "2026-11-06 09:02:00",
       "started_by": "88",
       "end_date": "2026-11-06 09:17:00",
       "ended_by": "88",
       "source": "API",
       "supervised": "0",
       "created_from": null,
       "client_work_order_id": null,
       "client_approved_by": null,
       "client_approved_date": null,
       "client_approval": null,
       "client_approval_date": null,
       "client_approval_note": null,
       "stock_branch_id": null,
       "schedule_id": null,
       "customer_name": "DMC (Londonist)",
       "service_name": "Room Cleaning",
       "branch_name": "Highbury Residence",
       "room_name": "Gold Ensuite Standard View Mid Level (G.04)",
       "room_type_id": "9",
       "room_type_name": "Gold Ensuite",
       "assigned_name": "Nikhil Test HK",
       "assigned_photo": "u88.jpg",
       "mopuan_employee_id": "345",
       "mopuan_supervisor_id": "101",
```

```
      "video_type": "r2",
      "video_before_type": "r2",
      "prefix_video": "https://<cdn-host>/",
      "prefix_video_before": "https://<cdn-host>/",
      "products": [
        {"id": "801", "company_id": "7", "work_order_id": "3492", "price": "100.00",
          "currency": "1", "qty": "2", "item_id": "41", "vat": "20.00",
          "added_date": "2026-11-05 16:40:12", "added_by": "-1", "build_id": null,
          "package_action": null, "settled_at": null}
      ],
      "notes": [
        {"id": "501", "company_id": "7", "work_order_id": "3492",
          "note": "Guest checks in at 14:00", "added_by": "87",
          "added_date": "2026-11-06 08:12:00", "status": "1", "visibility": "internal",
          "author_type": "staff", "added_by_name": "Jane Smith"}
      ]
    }
  ],
  "has_more": true,
  "next_before_id": 3473
 }
}
```


Fields of each work order:

Field	Notes
`id`	Work order id.
`company_id`	Company id.
`customer_id`	Customer id.
`branch`	Building id (the property).
`room`	Room id.
`service`	Service id.
`assigned_to`	OpSense user id of the assignee.
`assigned_by`	OpSense user id of who assigned it.
`assigned_date`	As stored. Not converted.
`appoinment_date`	Appointment. Converted to company time zone. Note the spelling.
`added_date`	When it was created. Converted.
`added_by`	User id of the creator. `-1` means created through the API.
`status`	1 pending, 2 in progress, 3 completed.
`video_before`	Before video, without prefix. `null` if none.

`video`	After (proof) video, without prefix. `null` if none.
`priority`	1 normal, 2 medium, 3 high, 4 urgent.
`is_vip`	0 or 1.
`start_date`	When work started. Converted.
`started_by`	User id who started it.
`end_date`	When work ended. Converted.
`ended_by`	User id who ended it.
`source`	Where the work order came from, for example `API`.
`supervised`	0 or a positive number. Not 0 means supervised.
`created_from`	As stored.
`client_work_order_id`, `client_approved_by`, `client_approved_date`, `client_approval`, `client_approval_date`, `client_approval_note`	Customer approval data, as stored. Dates are not converted.
`stock_branch_id`	Source location of the materials. `null` means the central warehouse.
`schedule_id`	Recurring schedule id, as stored.
`customer_name`	The customer's company name, or the contact name when there is no company name. `null` if the work order has no customer.
`service_name`	Name of the service.
`branch_name`	Name of the building.
`room_name`	Name of the room.
`room_type_id`	Room type id of the room.
`room_type_name`	Room type name.
`assigned_name`	First name and surname of the assignee.
`assigned_photo`	Profile picture file name of the assignee.
`mopuan_employee_id`	Mopuan id of the assignee.

`mopuan_supervisor_id`	Mopuan id of the assignee's first supervisor. `null` if none.
`video_type`, `video_before_type`	`r2` or `server`. `null` if no video.
`prefix_video`, `prefix_video_before`	Put in front of `video` / `video_before` to get the full URL. Empty if no video.
`products`	Product lines, see below.
`notes`	Notes that are not deleted, newest first, see below.

All joined names are `null` when the id points to nothing in this company.

Fields of each `products` line:

Field	Notes
`id`	Line id.
`company_id`, `work_order_id`	Ids.
`item_id`	Item id (`SRVGetProducts`).
`qty`	Quantity.
`price`	Line price: unit price times `qty`, taken when the line was saved.
`vat`	VAT value of the item when the line was saved.
`currency`	Currency id, as stored.
`added_date`	Converted.
`added_by`	`-1` for lines saved through the API.
`build_id`	Stock build id for package lines.
`package_action`	`assemble`, `deliver` or `release` for package lines, else `null`.
`settled_at`	When the line was settled in stock. Not converted.

Fields of each `notes` entry are the same as in section 4.1. Note dates are not converted.

Paging fields:

Field	Notes
`has_more`	Only when `limit` is sent. `true` if older work orders exist.

`next_before_id`	Only when `limit` is sent. `null` when `has_more` is `false`.

Date conversion: `appoinment_date`, `added_date`, `start_date`, `end_date` and the products' `added_date` are converted to the company time zone. The notes' `added_date` and the filter parameters are not.

`error_code`	Meaning
10015, 10016, 10099	Common errors, see section 1.2.

A request that matches nothing returns `"data": []`, not an error.

3.2 SRVAddWorkOrder

Creates a work order, or edits one when `edit_id` is sent. On edit, only the fields you send change. It also saves the product list, reserves or releases stock, uploads the before and after videos, and notifies people.

Field	Required	Notes
`api_key`	**Required**	See section 1.3.
`edit_id`	**Edit only**	Id of the work order to edit. Leave out to create.
`customer`	**Create**	Customer id (`SRVGetCustomers`). Not checked on create: always send it.
`property`	**Create**	Building id (`SRVGetProperties`). Not checked on create: always send it.
`room_flat`	**Create**	Room id. Rooms are listed under each building in `SRVGetProperties`. Not checked on create.
`service`	**Create**	Service id (`SRVGetServices`). Not checked on create.
`appointment_date`	**Create**	`YYYY-MM-DD HH:MM:SS`, company time zone. Converted to server time. Not checked on create. On edit, a different value reschedules.
`assigned_to`	Optional	OpSense user id of the assignee.
`assigned_by`	Optional	OpSense user id of who assigns.
`priority`	Optional	1 to 4. Default 1 on create.
`status`	Optional	1, 2 or 3. Default 1 on create.
`is_vip`	Optional	0 or 1. On create, not sent means 0.
`supervised`	Optional	A number above 0 sets it. 0 or not sent changes nothing, so it cannot be cleared here.

`start_date`, `end_date`	Optional	Saved as sent. Not converted: send server time (Europe/London).
`started_by`, `ended_by`	Optional	OpSense user ids.
`products`	Optional	JSON text, see below. On edit, sending it replaces the whole product list.
`stock_branch_id`	Optional	Source location for the products, from `SRVGetStockLocations`. `0` (or any number below 1) means the central warehouse. Leave out to keep the current location (central on create).
`send_notification_customer`	Optional	Any value other than empty or `0` sends the customer the confirmation email when an assignee is set (on create or reassign).
`form_source`	Optional	Free-text label saved as `source`. Default `API` on create.
`video_before`	Optional	Video file, `multipart/form-data`. Saved as `video_before`, and `start_date` is set to the time of the upload.
`video_after`	Optional	Video file, `multipart/form-data`. Saved as `video`, and `end_date` is set to the time of the upload.

`products` is a JSON text in one form field:

```
{"products": [
 {"item": 41, "qty": 2},
 {"item": 57, "qty": 1, "action": "assemble"}
]}
```

Key	Notes
`item`	Item id from `SRVGetProducts`. Items of another company or unknown items are skipped.
`qty`	Quantity. Lines with 0 or less are skipped.
`action`	Packages only (`is_package` = 1 in `SRVGetProducts`): `assemble` (default), `deliver` or `release`. Any other value counts as `assemble`.

If the same `item` appears twice, the last line wins. On edit, `{"products": []}` removes every line and gives the reserved stock back. An empty `products` field changes nothing.

Stock: products sent on create or edit are saved, and their stock is reserved from the source location. A work order saved with status 3 (completed) uses the stock up. If stock is short, nothing is saved (not even the other fields) and you get `10030` with the short items in `shortfalls`. Uploaded videos of a refused save are removed.

Who is notified, after a successful save:

Save	Assignee gets	Customer gets
Create with `assigned_to`	"New Work Order Assigned to You"	Confirmation email, if `send_notification_customer` is set and the customer has a contact email
Edit with a different `assigned_to`	"New Work Order Assigned to You" (the new assignee)	Same as above
Edit with a different `appointment_date`, same assignee	"Work Order Rescheduled"	Nothing
Any other edit, including sending the same appointment back	Nothing	Nothing

The assignee gets an OpSense notification, an email (if the user has an email address) and a Mopuan push (if the user is linked to Mopuan and the company has Mopuan credentials). A work order without an assignee notifies nobody.

Other side effects:

- On create, every active admin (role 1) of the company gets the OpSense notification "New Work Order Added via API".
- On create, and on edit when the status changes, the company's active webhooks receive `{"event": ..., "work_order_id": ...}` with the event `work_order_pending`, `work_order_in_progress` or `work_order_completed` for the new status.
- A failed email, push or webhook never turns a successful save into an error.

Create:

```
curl -X POST "https://<api-host>/v1/SRVAddWorkOrder" \
 -d "api_key=<api_key>" \
 -d "customer=55" -d "property=12" -d "room_flat=301" -d "service=7" \
 -d "appointment_date=2026-11-06 14:00:00" \
 -d "assigned_to=88" \
 -d "stock_branch_id=0" \
 -d "send_notification_customer=1" \
 --data-urlencode 'products={"products":[{"item":41,"qty":2},{"item":57,"qty":1,"action":"assemble"}]}'
```

Reassign and reschedule:

```
curl -X POST "https://<api-host>/v1/SRVAddWorkOrder" \
 -d "api_key=<api_key>" \
 -d "edit_id=3498" \
 -d "assigned_to=90" \
 -d "appointment_date=2026-11-07 09:30:00"
```

Upload the after video and complete:

```
curl -X POST "https://<api-host>/v1/SRVAddWorkOrder" \
 -F "api_key=<api_key>" \
 -F "edit_id=3498" \
```

```
 -F "status=3" \
 -F "video_after=@proof.mp4"
```

Success returns the saved work order row, with its video fields and `products`:

```
{
 "result": {
   "result": "1",
   "data": {
     "id": "3498",
     "company_id": "7",
     "customer_id": "55",
     "branch": "12",
     "room": "301",
     "service": "7",
     "assigned_to": "88",
     "appoinment_date": "2026-11-06 14:00:00",
     "status": "1",
     "priority": "1",
     "is_vip": "0",
     "source": "API",
     "stock_branch_id": null,
     "video": null,
     "video_before": null,
     "video_type": null,
     "video_before_type": null,
     "prefix_video": "",
     "prefix_video_before": "",
     "products": [
       {"id": "802", "work_order_id": "3498", "item_id": "41", "qty": "2",
         "price": "100.00", "vat": "20.00", "package_action": null}
     ]
   }
 }
}
```

The `data` row has every work order column listed in section 3.1 (it does not have the joined names such as `customer_name`), plus `video_type`, `video_before_type`, `prefix_video`, `prefix_video_before` and `products`. The dates `added_date`, `appoinment_date`, `start_date` and `end_date` and each product's `added_date` are converted to the company time zone.

In rare cases the save succeeds but the read-back fails. You then get only `{"result": "1", "data": {"id": "3498"}}`. Treat it as success and load the work order with `SRVGetWorkOrders` and `reference_id`. Never repeat the create.

error_code	Meaning
10017	Save failed.
10018	`edit_id` is not a work order of this company, or `status` is not 1, 2 or 3 (`error_text` says which). Ids of another company in `customer`, `property`, `room_flat`, `service`, `assigned_to`, `assigned_by`, `started_by` or `ended_by` are logged today and not yet refused, but may be refused with this code in a later release.
10019	`priority` is not 1 to 4.

10020	Video upload failed or the file is not a video. `error_text` has the reason.
10021	`products` is not valid JSON, or has no `products` list.
10022	`stock_branch_id` is not an active stock location of this company.
10030	Not enough stock, or the stock could not be reserved. `shortfalls` lists the short items, each with `item_name`, `requested` and `available` (packages can add `depth`). `shortfalls` can be empty when the stock could not be reserved for another reason.
10015, 10016, 10099	Common errors, see section 1.2.

3.3 SRVDeleteWorkOrder

Deletes a work order for good (a hard delete, not a soft delete). Its product lines and its notes are deleted with it, and the stock it still holds (package builds and reserved items) is given back. Stock that was already used up stays used.

Field	Required	Notes
`api_key`	**Required**	See section 1.3.
`work_order_id`	**Required**	Id of the work order. An id that does not exist, or belongs to another company, gives 10018.

```
curl -X POST "https://<api-host>/v1/SRVDeleteWorkOrder" \
 -d "api_key=<api_key>" \
 -d "work_order_id=3498"
```

```
{"result": {"result": "1", "result_text": "Data has been removed successfully."}}
```

error_code	Meaning
10018	The work order does not exist or is not a work order of this company.
10019	The delete failed. Nothing was changed.
10031	The reserved stock could not be given back. Nothing was changed. `error_text` has the reason.
10015, 10016, 10099	Common errors, see section 1.2.

No webhook and no notification is sent for a delete.

4. Work order notes

A note belongs to a work order and to the OpSense user who wrote it (`added_by`). Only the author can edit or delete a note: send the author's OpSense user id as `user_id`. Use the `user_id` from `SRVGetUserRoles`.

Note `status` values:

`status`	Meaning
1	Added, not edited.
2	Edited.
0	Deleted. Deleted notes are not returned.

Note dates are not converted: `added_date` is in server time (Europe/London).

4.1 SRVGetWorkOrderNotes

Returns the notes of one work order, newest first. Deleted notes are left out.

Field	Required	Notes
`api_key`	**Required**	See section 1.3.
`work_order_id`	**Required**	Id of the work order.

```
curl -X POST "https://<api-host>/v1/SRVGetWorkOrderNotes" \
 -d "api_key=<api_key>" \
 -d "work_order_id=3492"
```

```
{
 "result": {
   "result": "1",
   "data": [
     {"id": "501", "company_id": "7", "work_order_id": "3492",
       "note": "Guest checks in at 14:00", "added_by": "87",
       "added_date": "2026-11-06 08:12:00", "status": "1", "visibility": "internal",
       "author_type": "staff", "added_by_name": "Jane Smith"}
   ]
 }
}
```

Field	Notes
`id`	Note id.
`company_id`	Company id.
`work_order_id`	Work order id.
`note`	Note text. Plain text.
`added_by`	OpSense user id of the author.

`added_date`	When the note was added, or last edited. Not converted.
`status`	1 or 2, see above.
`visibility`	`internal` or `customer`. Notes added through this API are `internal`.
`author_type`	`staff` or `customer`. Notes added through this API are `staff`.
`added_by_name`	First name and surname of the author. `null` if the author is not a user of this company.

`error_code`	Meaning
10018	The work order does not exist or is not a work order of this company.
10015, 10016, 10099	Common errors, see section 1.2.

4.2 SRVAddWorkOrderNote

Adds a note. The work order's assignee is told about it.

Field	Required	Notes
`api_key`	**Required**	See section 1.3.
`work_order_id`	**Required**	Id of the work order.
`note`	**Required**	Note text.
`user_id`	Optional	OpSense user id of the author. The service does not require it, but without it the note is saved with no author. Always send it.

```
curl -X POST "https://<api-host>/v1/SRVAddWorkOrderNote" \
 -d "api_key=<api_key>" \
 -d "work_order_id=3492" \
 -d "user_id=87" \
 --data-urlencode "note=Guest checks in at 14:00"
```

Success returns the saved note, with the fields of section 4.1 and `status` 1:

```
{"result": {"result": "1", "data": {"id": "501", "work_order_id": "3492",
 "note": "Guest checks in at 14:00", "added_by": "87", "added_date": "2026-11-06 08:12:00",
 "status": "1", "added_by_name": "Jane Smith"}}}
```

Notification: when the work order has an assignee, the author is not the assignee, and the assignee has an email address, the assignee gets an OpSense notification "New Note on Work Order" and an email with the note text. There is no Mopuan push. A failure to notify never fails the save.

`error_code`	Meaning

10017	Save failed.
10018	The work order does not exist or is not a work order of this company. A `user_id` of another company is logged today and not yet refused.
10019	`note` is empty.
10015, 10016, 10099	Common errors, see section 1.2.

4.3 SRVUpdateWorkOrderNote

Changes the text of a note. Only the author can do this. The note gets `status` 2 and its `added_date` becomes the time of the edit.

Field	Required	Notes
`api_key`	**Required**	See section 1.3.
`note_id`	**Required**	Id of the note.
`user_id`	**Required**	OpSense user id of the author. Must match the note's `added_by`.
`note`	**Required**	New text.

```
curl -X POST "https://<api-host>/v1/SRVUpdateWorkOrderNote" \
 -d "api_key=<api_key>" \
 -d "note_id=501" \
 -d "user_id=87" \
 --data-urlencode "note=Guest checks in at 15:00"
```

Success returns the updated note, with the fields of section 4.1:

```
{"result": {"result": "1", "data": {"id": "501", "work_order_id": "3492",
 "note": "Guest checks in at 15:00", "added_by": "87", "added_date": "2026-11-06 09:30:00",
 "status": "2", "added_by_name": "Jane Smith"}}}
```

`error_code`	Meaning
10017	Save failed.
10018	The note does not exist, is not a note of this company, or `user_id` is not its author.
10019	`note` is empty.
10015, 10016, 10099	Common errors, see section 1.2.

The service does not check whether the note was deleted: editing a deleted note makes it visible again (`status` 2).

4.4 SRVDeleteWorkOrderNote

Deletes a note. Only the author can do this. It is a soft delete: the note gets `status` 0 and is no longer returned.

Field	Required	Notes
`api_key`	**Required**	See section 1.3.
`note_id`	**Required**	Id of the note.
`user_id`	**Required**	OpSense user id of the author. Must match the note's `added_by`.

```
curl -X POST "https://<api-host>/v1/SRVDeleteWorkOrderNote" \
 -d "api_key=<api_key>" \
 -d "note_id=501" \
 -d "user_id=87"
```

```
{"result": {"result": "1", "result_text": "Data has been removed successfully."}}
```

`error_code`	Meaning
10018	The note does not exist, is not a note of this company, or `user_id` is not its author.
10019	Nothing was changed. This is also the answer when the note is already deleted.
10015, 10016, 10099	Common errors, see section 1.2.

5. Lookups for the form

All lookups are read-only, take `api_key`, and return `{"result": "success", "data": ...}`. Except `SRVGetStockLocations` and `SRVGetTimezone`, they also take an optional `reference_id` to return one row. Every row is the table row as stored, so ids and numbers may arrive as strings.

5.1 SRVGetCustomers

Customer list for the "Customer" field.

Field	Required	Notes
`api_key`	**Required**	See section 1.3.
`reference_id`	Optional	Return only this customer.

```
curl -X POST "https://<api-host>/v1/SRVGetCustomers" \
 -d "api_key=<api_key>"
```

```
{
 "result": {
   "result": "success",
   "data": [
     {"id": "55", "company_id": "7", "company_name": "DMC (Londonist)",
       "contact_name": "Tom Hale", "contact_email": "tom@example.com",
       "contact_phone": "+44 20 7946 0000", "contact_title": "Manager",
       "address": "1 High Street, London", "sector": "Hospitality",
       "company_logo": "customer_65a.png", "pp": null, "source": "API", "status": "1",
       "added_date": "2026-02-03 10:00:00", "added_by": "-1", "is_deleted": "0",
       "deleted_at": null, "deleted_by": null, "language": "en", "is_internal": "0"}
   ]
 }
}
```


Field	Notes
`id`	Customer id. Send it as `customer` in `SRVAddWorkOrder`.
`company_id`	Company id.
`company_name`	Customer company name.
`contact_name`, `contact_email`, `contact_phone`, `contact_title`	Contact person.
`address`	Address.
`sector`	Business sector.
`company_logo`	Logo file name.
`pp`	Profile picture file name, as stored.
`source`	Where the customer came from, for example `API`.
`status`	Status, as stored.
`added_date`	Converted to the company time zone.
`added_by`	User id of the creator. `-1` means the API.
`is_deleted`	`1` if the customer was deleted. Deleted customers are still listed: do not offer them in the form.
`deleted_at`, `deleted_by`	Delete time (not converted) and user.
`language`	Customer language code.
`is_internal`	`1` if this customer stands for the company itself.

`password`	Also present in the row. Not for the app: never show or store it.

`error_code`	Meaning
10015, 10016, 10099	Common errors, see section 1.2.

5.2 SRVGetProperties

Buildings (properties), each with its rooms. For the "Building" and "Room" fields and the project cards.

Field	Required	Notes
`api_key`	**Required**	See section 1.3.
`reference_id`	Optional	Return only this building.

```
curl -X POST "https://<api-host>/v1/SRVGetProperties" \
 -d "api_key=<api_key>"
```


```
{
 "result": {
   "result": "success",
   "data": [
     {
       "id": "12", "company_id": "7", "customer_id": "55", "name": "Highbury Residence",
         "address": "10 Holloway Road, London", "lng": "-0.1030", "lat": "51.5465",
       "contact_name": "Front desk", "contact_email": "desk@example.com",
         "contact_phone": "+44 20 7946 0001",
       "added_date": "2026-02-03 10:05:00", "added_by": "-1", "status": "1",
         "logo": "branch_65b.jpg",
       "branch_type": null, "is_stock_location": "0", "stock_notes": null,
         "mopuan_qr_version": "0",
       "room_flat_data": [
         {"id": "301", "company_id": "7", "branch_id": "12", "name": "G.04",
           "room_image": null, "added_date": "2026-02-03 10:10:00", "added_by": "-1",
           "status": "1", "room_type_id": "9", "room_type_name": "Gold Ensuite"}
       ]
     }
   ]
 }
}
```


Field	Notes
`id`	Building id. Send it as `property` in `SRVAddWorkOrder`.
`company_id`	Company id.
`customer_id`	Owner customer id.
`name`	Building name.
`address`	Address.

`lng`, `lat`	Coordinates, as text.
`contact_name`, `contact_email`, `contact_phone`	Contact person.
`added_date`	Converted to the company time zone.
`added_by`	User id of the creator. `-1` means the API.
`status`	Status, as stored (1 active). Buildings of every status are listed.
`logo`	Logo file name.
`branch_type`	Only meaningful for the company's own locations.
`is_stock_location`	`1` if stock may be held here.
`stock_notes`	Stock notes, as stored.
`mopuan_qr_version`	Version counter for the room QR codes.
`room_flat_data`	The building's rooms, every status. `[]` if none.

Fields of each room in `room_flat_data`:

Field	Notes
`id`	Room id. Send it as `room_flat` in `SRVAddWorkOrder`.
`company_id`, `branch_id`	Company and building ids.
`name`	Room name.
`room_image`	Image file name.
`added_date`	Converted.
`added_by`	User id of the creator.
`status`	Status, as stored.
`room_type_id`	Room type id.
`room_type_name`	Room type name. `null` if none.

`error_code`	Meaning
10015, 10016, 10099	Common errors, see section 1.2.

5.3 SRVGetRoomTypes

Room types, for the room form.

Field	Required	Notes
`api_key`	**Required**	See section 1.3.
`reference_id`	Optional	Return only this room type.

```
curl -X POST "https://<api-host>/v1/SRVGetRoomTypes" \
 -d "api_key=<api_key>"
```

```
{"result": {"result": "success", "data": [{"id": "9", "company_id": "7",
 "name": "Gold Ensuite", "added_date": "2026-02-03 10:08:00", "added_by": "-1",
 "status": "1"}]}}
```

Field	Notes
`id`	Room type id. Send it as `room_type_id` in `SRVAddRoomFlat`.
`company_id`	Company id.
`name`	Room type name.
`added_date`	Converted to the company time zone.
`added_by`	User id of the creator.
`status`	Status, as stored. Room types of every status are listed.

`error_code`	Meaning
10015, 10016, 10099	Common errors, see section 1.2.

5.4 SRVGetServices

Services, for the "Service" field.

Field	Required	Notes
`api_key`	**Required**	See section 1.3.
`reference_id`	Optional	Return only this service.

```
curl -X POST "https://<api-host>/v1/SRVGetServices" \
 -d "api_key=<api_key>"
```

{"result": {"result": "success", "data": [{"id": "4", "company_id": "7",
 "name": "Room Cleaning", "added_date": "2026-01-10 09:00:00", "added_by": "3",
 "status": "1"}]}}

Field	Notes
`id`	Service id. Send it as `service` in `SRVAddWorkOrder`.
`company_id`	Company id.
`name`	Service name.
`added_date`	Converted to the company time zone.
`added_by`	User id of the creator.
`status`	Status, as stored. Services of every status are listed.

`error_code`	Meaning
10015, 10016, 10099	Common errors, see section 1.2.

5.5 SRVGetProducts

Inventory items, for the "Products" part of the work order form.

Field	Required	Notes
`api_key`	**Required**	See section 1.3.
`reference_id`	Optional	Return only this item.

```
curl -X POST "https://<api-host>/v1/SRVGetProducts" \
 -d "api_key=<api_key>"
```


```
{
 "result": {
   "result": "success",
   "data": [
     {"id": "41", "company_id": "7", "item_name": "Towel set", "category": "Linen",
       "unit": "pcs", "quantity_available": "120.00", "location": "Main store",
       "last_updated": "2026-11-05 12:00:00", "supplier_name": "Linen Ltd",
       "unit_price": "50.00", "vat": "20.00", "total_value": "6000.00",
       "status": "In Stock", "restock_date": null, "notes": null, "sku": "TWL-01",
       "min_stock": "20.00", "qty_reserved": "4.00", "barcode": null, "brand": null,
       "model": null, "weight": null, "dimensions": null, "supplier_id": null,
       "lead_time_days": null, "reorder_qty": "0.00", "image": null, "track_batches": "0",
       "is_package": "0"}
   ]
 }
}
```

Field	Notes
`id`	Item id. Send it as `item` in the `products` JSON of `SRVAddWorkOrder`.
`company_id`	Company id.
`item_name`	Item name.
`category`, `unit`, `brand`, `model`, `sku`, `barcode`, `notes`	Descriptive fields.
`quantity_available`	Quantity in stock, as stored.
`qty_reserved`	Quantity reserved by work orders and deliveries.
`location`	Free-text location.
`unit_price`	Price of one unit. A work order line price is this times `qty`.
`vat`	VAT value.
`total_value`	`quantity_available` times `unit_price`.
`status`	`In Stock`, `Low Stock`, `Out of Stock` or `Discontinued`.
`min_stock`, `reorder_qty`, `lead_time_days`, `supplier_id`, `supplier_name`	Stock planning fields.
`weight`, `dimensions`, `image`	Item details.
`track_batches`	`1` if stock is tracked in batches.
`is_package`	`1` if the item is a package (assembled from other items). Packages take an `action` in the `products` JSON.
`last_updated`, `restock_date`	Dates as stored, in server time. Not converted.

All items are listed, whatever their `status`. Free stock for an item is not returned as one number: use `quantity_available` and `qty_reserved`.

error_code	Meaning
10015, 10016, 10099	Common errors, see section 1.2.

5.6 SRVGetStockLocations

The "Source location" list of the work order form.

Field	Required	Notes
`api_key`	**Required**	See section 1.3.

```
curl -X POST "https://<api-host>/v1/SRVGetStockLocations" \
 -d "api_key=<api_key>"
```

```
{
 "result": {
   "result": "success",
   "data": [
     {"id": null, "name": "Central warehouse", "group": "central", "customer_name": ""},
     {"id": 14, "name": "Leyton Store", "group": "internal", "customer_name": "Londonist"},
     {"id": 31, "name": "Highbury Residence", "group": "external",
       "customer_name": "DMC (Londonist)"}
   ]
 }
}
```

Field	Notes
`id`	Send as `stock_branch_id`. `null` is the central warehouse: send `0`, or leave the field out on create.
`name`	Location name.
`group`	`central`, `internal` (own locations) or `external` (stock kept at a client site). Use it for the section headings.
`customer_name`	Owner of the location. Show it before the name for `external` rows.

The central warehouse is always first, then the internal locations, then the external ones. Only active locations a work order can be saved with are listed.

`error_code`	Meaning
10015, 10016, 10099	Common errors, see section 1.2.

5.7 SRVGetTimezone

The company's time zone. Use it to show and enter dates.

Field	Required	Notes
`api_key`	**Required**	See section 1.3.

```
curl -X POST "https://<api-host>/v1/SRVGetTimezone" \
 -d "api_key=<api_key>"
```

```
{"result": {"result": "success", "data": {"timezone": "Europe/London"}}}
```

Field	Notes
`timezone`	Time zone name. `Europe/London` when the company has none set.

`error_code`	Meaning
10015, 10016, 10099	Common errors, see section 1.2.

6. Creating a customer from the form

The new work order form can create a customer, then a building, then a room, in three calls. Each call creates when `edit_id` is not sent, and edits when it is.

None of these services checks that a field is filled. A field that is not sent is saved as an empty value. On **edit**, every field the form shows is rewritten, so send all of them again, not only the ones that changed. If an edit changes nothing, the service answers 10017.

Images are optional and are sent as the form file `file` (`multipart/form-data`). Only `.jpg`, `.jpeg`, `.png` and `.webp` files are accepted. A file with another extension is skipped without an error. The response gives the stored file name only.

6.1 SRVAddCustomer

Creates or edits a customer.

Field	Required	Notes
`api_key`	**Required**	See section 1.3.
`edit_id`	**Edit only**	Id of the customer to edit. Leave out to create.
`company_name`	Optional	Company name.
`sector`	Optional	Business sector.
`contact_name`	Optional	Contact person.
`contact_email`	Optional	Contact email. Work order confirmation emails go here.
`contact_phone`	Optional	Contact phone.
`contact_title`	Optional	Contact title.
`address`	Optional	Address.
`status`	Optional	Send `1` for an active customer. A value that is not sent is saved empty.
`file`	Optional	Logo image.

`form_source`	Optional	Free-text label saved as `source`. Default `API` on create.

```
curl -X POST "https://<api-host>/v1/SRVAddCustomer" \
 -F "api_key=<api_key>" \
 -F "company_name=Riverside Hotels" \
 -F "contact_name=Ana Lopez" \
 -F "contact_email=ana@example.com" \
 -F "contact_phone=+44 20 7946 0100" \
 -F "status=1" \
 -F "file=@logo.png"
```

Success returns the saved customer row, with the fields of section 5.1:


```
{"result": {"result": "1", "data": {"id": "812", "company_id": "7",
 "company_name": "Riverside Hotels", "contact_name": "Ana Lopez",
 "contact_email": "ana@example.com", "contact_phone": "+44 20 7946 0100", "status": "1",
 "source": "API", "company_logo": "customer_65c.png", "added_date": "2026-11-06 10:15:00",
 "added_by": "-1", "is_deleted": "0"}}}
```


On create, every active admin (role 1) gets the OpSense notification "New Customer Registered via API". `added_date` is converted to the company time zone. On edit, the old logo file is kept when a new one is uploaded.

`error_code`	Meaning
10017	Save failed, or an edit changed nothing.
10018	`edit_id` is not a customer of this company.
10015, 10016, 10099	Common errors, see section 1.2.

6.2 SRVAddProperty

Creates or edits a building.

Field	Required	Notes
`api_key`	**Required**	See section 1.3.
`edit_id`	**Edit only**	Id of the building to edit. Leave out to create.
`customer_id`	Optional	Owner customer id (`SRVGetCustomers`). Send it: the service does not require it. An id of another company is logged today and not yet refused.
`name`	Optional	Building name.
`contact_name`	Optional	Contact person.
`contact_email`	Optional	Contact email.
`contact_phone`	Optional	Contact phone.

`address`	Optional	Address.
`lat`, `lng`	Optional	Coordinates.
`status`	Optional	Default 1 (also on edit).
`file`	Optional	Logo image. On edit, a new logo replaces the old one and the old file is deleted.

```
curl -X POST "https://<api-host>/v1/SRVAddProperty" \
 -F "api_key=<api_key>" \
 -F "customer_id=812" \
 -F "name=Riverside North" \
 -F "address=5 Quay Street, London"
```

Success returns the saved building row, with the building fields of section 5.2 (without `room_flat_data`):

```
{"result": {"result": "1", "data": {"id": "130", "company_id": "7", "customer_id": "812",
 "name": "Riverside North", "address": "5 Quay Street, London", "status": "1", "logo": null,
 "is_stock_location": "0", "added_date": "2026-11-06 10:20:00", "added_by": "-1"}}}
```

On create, every active admin (role 1) gets the OpSense notification "New Branch Has Been Added via API". `added_date` is converted. A new building is not a stock location.

`error_code`	Meaning
10017	Save failed, or an edit changed nothing.
10018	`edit_id` is not a building of this company.
10015, 10016, 10099	Common errors, see section 1.2.

6.3 SRVAddRoomFlat

Creates or edits a room (flat) of a building.

Field	Required	Notes
`api_key`	**Required**	See section 1.3.
`edit_id`	**Edit only**	Id of the room to edit. Leave out to create.
`property`	**Required**	Building id. Also required on edit.
`name`	Optional	Room name.
`room_type_id`	Optional	Room type id (`SRVGetRoomTypes`). Not sent: unchanged on edit, no type on create.
`status`	Optional	Default 1 (also on edit).

`file`	Optional	Room image. On edit, a new image replaces the old one and the old file is deleted.

```
curl -X POST "https://<api-host>/v1/SRVAddRoomFlat" \
 -F "api_key=<api_key>" \
 -F "property=130" \
 -F "name=1.01" \
 -F "room_type_id=9"
```

Success returns the saved room, with the room fields of section 5.2 and `room_type_name`:

```
{"result": {"result": "1", "data": {"id": "950", "company_id": "7", "branch_id": "130",
 "name": "1.01", "room_image": null, "status": "1", "room_type_id": "9",
 "added_date": "2026-11-06 10:25:00", "added_by": "-1", "room_type_name": "Gold Ensuite"}}}
```

`added_date` is converted. The image is stored before `property` and `room_type_id` are checked, so a refused request with a file leaves the file behind on the server.

error_code	Meaning
10017	Save failed, or an edit changed nothing.
10018	`edit_id` is not a room of this company.
10019	`property` is missing. `error_text` is "You don't have access this resource. Please check your api_key."
10028	`property` is not a building of this company.
10029	`room_type_id` is not a room type of this company.
10015, 10016, 10099	Common errors, see section 1.2.

7. Notifications

7.1 SRVGetNotifications

Returns notifications, newest first (highest id first). Panic alerts are notifications whose `title` starts with `PANIC BUTTON - `.

Field	Required	Notes
`api_key`	**Required**	See section 1.3.
`mopuan_user_id`	Optional	Only this employee's notifications. Your Mopuan employee id.
`user_id`	Optional	The same, by OpSense user id. Ignored if `mopuan_user_id` is sent.

`limit`	Optional	Page size, a number above 0. Adds `has_more` and `next_before_id`.
`before_id`	Optional	Only notifications with an id lower than this. Send the `next_before_id` of the previous page.

Without `mopuan_user_id` and `user_id` you get the notifications of the whole company. Always send one of them. Without `limit` you get every notification and no paging fields.

Paging works as for `SRVGetWorkOrders` (section 3.1).

```
curl -X POST "https://<api-host>/v1/SRVGetNotifications" \
 -d "api_key=<api_key>" \
 -d "mopuan_user_id=345" \
 -d "limit=30"
```

```
{
 "result": {
   "result": "success",
   "data": [
     {
       "id": "25510",
       "user_id": "87",
       "mopuan_user_id": "345",
       "link": "https://<portal-host>/calendar",
       "title": "New Note on Work Order",
       "description": "New note added by Jane Smith",
       "added_date": "2026-11-06 14:02:05",
       "is_read": "0",
       "read_date": null
     }
   ],
   "has_more": true,
   "next_before_id": 25481
 }
}
```

Field	Notes
`id`	Notification id. A newer notification always has a higher id.
`user_id`	OpSense user id of the employee the notification is for.
`mopuan_user_id`	Mopuan employee id of the same employee.
`link`	Portal page the notification is about.
`title`	Short title. Plain text.
`description`	One-line description. Plain text.
`added_date`	Converted to the company time zone.
`is_read`	`1` if the employee has read it, in the app or in the portal. `0` if not.

`read_date`	When it was read, converted. `null` if unread.
`has_more`	Only when `limit` is sent. `true` if older notifications exist.
`next_before_id`	Only when `limit` is sent. `null` when `has_more` is `false`.

Only notifications for staff users are returned. Read state is per employee and shared with the portal.

error_code	Meaning
10015, 10016, 10099	Common errors, see section 1.2.

7.2 SRVMarkNotificationsRead

Marks one notification, or all notifications of an employee, as read.

Field	Required	Notes
`api_key`	**Required**	See section 1.3.
`mopuan_user_id`	One of the two	The employee whose notifications are marked. Your Mopuan employee id.
`user_id`	One of the two	The same, by OpSense user id. Ignored if `mopuan_user_id` is sent.
`notification_id`	One of the two	Mark this notification as read.
`all`	One of the two	`1` marks every unread notification of the employee as read. Used only when `notification_id` is not sent.

```
curl -X POST "https://<api-host>/v1/SRVMarkNotificationsRead" \
 -d "api_key=<api_key>" \
 -d "mopuan_user_id=345" \
 -d "notification_id=25510"
```

```
curl -X POST "https://<api-host>/v1/SRVMarkNotificationsRead" \
 -d "api_key=<api_key>" \
 -d "mopuan_user_id=345" \
 -d "all=1"
```

```
{"result": {"result": "success", "updated": 12}}
```

Field	Notes
`updated`	Number of notifications that changed from unread to read. Marking a notification that is already read, or that belongs to someone else, returns `updated: 0`, not an error. You can safely repeat a call.

error_code	Meaning

10019	Neither `mopuan_user_id` nor `user_id` sent, or neither `notification_id` nor `all=1`.
10020	The employee is not an employee of this company.
10015, 10016, 10099	Common errors, see section 1.2.

8. Dashboard and staff

8.1 SRVGetServiceDurations

Average time from start to end of completed work orders, per service. For "Task durations" on the operations dashboard.

Field	Required	Notes
`api_key`	**Required**	See section 1.3.
`start_date`	Optional	`YYYY-MM-DD`, company time zone. Default: 120 days before `end_date`. A value that is not a date counts as not sent.
`end_date`	Optional	`YYYY-MM-DD`, company time zone. Default: today in the company time zone.

Work orders are selected by appointment date, from the start of `start_date` to the end of `end_date` in the company time zone (converted to server time). Only completed work orders (status 3) with a start and an end count, and only durations from 5 to 600 minutes. If `start_date` is later than `end_date` the list is empty.

For "Today", send today as both dates. For "This week", send Monday and today. To compare with yesterday, make a second call for yesterday.

```
curl -X POST "https://<api-host>/v1/SRVGetServiceDurations" \
 -d "api_key=<api_key>" \
 -d "start_date=2026-11-06" -d "end_date=2026-11-06"
```

```
{
 "result": {
   "result": "success",
   "start_date": "2026-11-06",
   "end_date": "2026-11-06",
   "data": [
     {"service_id": 4, "service_name": "Room Cleaning", "avg_minutes": 72, "samples": 31},
     {"service_id": 9, "service_name": "Delivery", "avg_minutes": 48, "samples": 6}
   ]
 }
}
```

Field	Notes
`start_date`, `end_date`	The dates used, after defaults.

`data`	One row per service, sorted by service name. A service with no qualifying work order is not listed.
`service_id`	Service id.
`service_name`	Service name. `null` if the service no longer exists.
`avg_minutes`	Average duration, rounded to whole minutes.
`samples`	How many work orders the average is based on.

`error_code`	Meaning
10015, 10016, 10099	Common errors, see section 1.2.

8.2 SRVGetWeeklyRoute

Who is on the weekly route (rota) on which day. For the calendar and "Staff status".

Field	Required	Notes
`api_key`	**Required**	See section 1.3.
`start_date`	Optional	`YYYY-MM-DD`.
`end_date`	Optional	`YYYY-MM-DD`.

The range filter works only when both dates are sent and valid. Both are compared as whole dates, so a range can cross a month. Without both, every route day is returned. Route days are calendar dates and are not converted.

```
curl -X POST "https://<api-host>/v1/SRVGetWeeklyRoute" \
 -d "api_key=<api_key>" \
 -d "start_date=2026-11-02" -d "end_date=2026-11-08"
```


```
{
 "result": {
   "result": "success",
   "data": [
     {
       "205": {
         "in_route": true,
         "routes": {
           "2026-11-02": "09:00 - 17:00",
           "2026-11-03": "Day Off"
         },
         "user_info": {
           "id": "205", "company_id": "1", "email": "a.smith@example.com",
             "phone": "+44 7700 900000",
           "name": "Alice", "surname": "Smith", "role": "5", "active": "1",
             "is_deleted": "0",
           "profile_picture": "u205.jpg", "only_assigned": "0", "mopuan_user_id": "345",
           "mopuan_beacon": null, "mopuan_beacon_type": null, "contract": "Full time",
```

```
            "duty": "Housekeeping"
          }
        },
        "208": {"in_route": false, "routes": [], "user_info": {"id": "208", "name": "Ben",
          "surname": "Jones"}}
      }
    ]
  }
}
```


`data` is a list with one element. That element is an object keyed by OpSense user id, with one entry for every active, not deleted user who is marked as a weekly route user. Users in the object come back even when they have no route day.

Field	Notes
`in_route`	`true` if the user has at least one route day in the range.
`routes`	Object keyed by date as stored: year, month and day joined by `-`, normally zero padded (`2026-11-02`). The value is the event, for example `Day Off`, or `start - end` (for example `09:00 - 17:00`) when the event is `Choose Time Range`. When the user has no route day it is an empty list `[]`, not an empty object.
`user_info`	The user: `id`, `company_id`, `email`, `phone`, `name`, `surname`, `role`, `active`, `is_deleted`, `profile_picture`, `only_assigned`, `mopuan_user_id`, `mopuan_beacon`, `mopuan_beacon_type`, `contract`, `duty`.

If the company has no weekly route users, `data` is `[[]]`.

`error_code`	Meaning
10015, 10016, 10099	Common errors, see section 1.2.

9. Roles

9.1 SRVGetUserRoles

Returns the employee's role, permissions and which employees they can see. For the module menu, for buttons and for counting work orders. Only active employees are returned.

Field	Required	Notes
`api_key`	**Required**	See section 1.3.
`mopuan_user_id`	Optional	Return only this employee. Your Mopuan employee id.
`user_id`	Optional	The same, by OpSense user id. Ignored if `mopuan_user_id` is sent.

Without either filter every active employee is returned.

```
curl -X POST "https://<api-host>/v1/SRVGetUserRoles" \
 -d "api_key=<api_key>" \
 -d "mopuan_user_id=345"
```


```
{
 "result": {
   "result": "success",
   "data": [
     {
       "user_id": 12,
       "mopuan_user_id": "345",
       "name": "Jane Doe",
       "only_assigned": false,
       "role": {
         "id": 5,
         "name": "Supervisor",
         "hierarchy_level": 2,
         "hierarchy_name": "Supervisor",
         "is_global": false,
         "active": true,
         "permissions": {
           "work-orders": {"view": true, "create": true, "edit": true, "delete": false},
           "users-list": {"view": false, "create": false, "edit": false, "delete": false}
         }
       },
       "supervisors": [{"user_id": 3, "mopuan_user_id": "101"}],
       "direct_reports": [{"user_id": 20, "mopuan_user_id": "410"}],
       "visibility": {"scope": "team", "user_ids": [12, 20, 30], "mopuan_user_ids": ["345",
         "410"]}
     }
   ],
   "locked_menus": ["kpi-dashboard", "work-order-ai-audit"]
 }
}
```


Field	Notes
`user_id`	OpSense user id. Use it as `user_id` in the note services.
`mopuan_user_id`	Mopuan employee id, or `null` if the employee is not linked to Mopuan.
`name`	First name and surname.
`only_assigned`	`true` if the portal limits this employee to their own data.
`role.id`, `role.name`	Role id and name.
`role.hierarchy_level`	0 and 1 see everyone, 2 sees their team, 3 sees themselves.
`role.hierarchy_name`	Super Admin, Admin, Supervisor or User.
`role.is_global`	`true` only for the built-in Admin role.
`role.active`	`false` if the role is switched off.

`role.permissions`	One entry per portal module, keyed by section key. Each has `view`, `create`, `edit` and `delete` as `true` or `false`. A module the role cannot use is listed with all four `false`.
`supervisors`	Who the employee reports to. Each has `user_id` and `mopuan_user_id`.
`direct_reports`	Employees who report to this employee. Same shape.
`visibility.scope`	`all`, `team` or `self`.
`visibility.user_ids`	OpSense ids the employee can see, including themselves. Not sent when `scope` is `all`.
`visibility.mopuan_user_ids`	The same as Mopuan ids. Employees not linked to Mopuan are left out. Not sent when `scope` is `all`.
`locked_menus`	Top-level field. Modules this company has switched off, as section keys. `[]` if none.

Show a module only when both are true:

1. `role.permissions[<key>].view` is `true`, and
2. `<key>` is not in `locked_menus`.

Section keys for the mobile menu:

Module	Section key
Work Order management	`work-orders`
AI Visor	`work-orders`, and `work-order-ai-audit` not in `locked_menus`
AI SPV	`work-orders`
User management	`users-list`
KPI	`kpi-dashboard`
Customer management	`customers`
Role management	`role-access`
Account & Device	`account-device-manager`
Beacon inventory, Reader inventory	`beacon-status-report`

Use `create`, `edit` and `delete` the same way for buttons: for example, show "New work order" only when `permissions["work-orders"].create` is `true`.

To count or list work orders for the logged-in user, use `visibility`: `all` sees every work order, `team` sees those whose `assigned_to` is in `visibility.user_ids`, `self` only their own.

error_code	Meaning
10020	A filter was sent and no active employee of this company matches. `error_text` is "User not found."
10015, 10016, 10099	Common errors, see section 1.2.

10. Panic button

10.1 SRVPanicButton

Sends a panic alert for an employee to their supervisors and to the company's upper management. Call it when the employee presses the panic button.

Field	Required	Notes
`api_key`	**Required**	See section 1.3.
`mopuan_user_id`	**Required**	Mopuan id of the employee who pressed the button. Must be an active employee of this company.
`location`	Optional	Free text, added to the alert as `Location: ...`.
`message`	Optional	Free text, added to the alert as `Message: ...`.

```
curl -X POST "https://<api-host>/v1/SRVPanicButton" \
 -d "api_key=<api_key>" \
 -d "mopuan_user_id=345" \
 -d "location=Highbury Residence, floor 3" \
 --data-urlencode "message=Need help now"
```

```
{
 "result": {
   "result": "1",
   "result_text": "Panic button alert sent successfully.",
   "notified_count": 2,
   "notified_users": [
     {"id": "3", "name": "Sam Carter", "email": "sam@example.com"},
     {"id": "4", "name": "Dana Wells", "email": "dana@example.com"}
   ],
   "triggered_at": "2026-11-06 14:31:08"
 }
}
```

Who receives it (each person once):

1. The employee's direct supervisors (active users).
2. Every active user of the company whose role has hierarchy level 2 or lower (Supervisor, Admin, Super Admin). The employee who pressed the button is also a recipient when their own role is in that group.

Channels, for each recipient:

- An OpSense notification with the title `PANIC BUTTON - <employee name>` and a description `<name> has triggered the panic button.` followed by `Location: ...` and `Message: ...` when sent. The link is the portal notifications page.
- A Mopuan push, when the company has Mopuan credentials and the recipient is linked to Mopuan.
- An email, when the recipient has an email address.

No webhook is sent.

Field	Notes
`result_text`	A fixed text.
`notified_count`	Number of recipients.
`notified_users`	The recipients: `id`, `name` and `email`.
`triggered_at`	When the alert was made, converted to the company time zone.

The call succeeds with `notified_count` 0 when there is nobody to notify. A failed email or push does not change the answer.

error_code	Meaning
10020	`mopuan_user_id` is empty.
10021	No active employee of this company has this `mopuan_user_id`.
10015, 10016, 10099	Common errors, see section 1.2.

11. AI Visor

These two services return the AI cleaning audits of the work orders. Each work order counts once, by its latest audit. The full field lists, labels, colours and verdicts are in the AI Visor API guide; this section lists the parameters and the main fields.

Both services answer `10040` when AI Visor is switched off for the company. Both take an optional `mopuan_employee_id` (only work orders assigned to that user) and an optional `lang` (`en` default, `tr`, `el` or `vi`) for the labels in the `display` object.

Dates you send are in the company time zone. Dates in the response are converted to it.

11.1 SRVGetAiVisorOverview

Totals, trend, buildings, top issues and a paged list of audited work orders. For the AI Visor overview screen.

Field	Required	Notes
`api_key`	**Required**	See section 1.3.

`start_date`	Optional	`YYYY-MM-DD`. Default: 30 days ago.
`end_date`	Optional	`YYYY-MM-DD`. Default: today. Swapped with `start_date` if earlier.
`branch_ids`	Optional	Comma-separated building ids. Default: all.
`verdict`	Optional	`all` (default), `approved`, `failed` or `review`. Filters the `audits` list only.
`page`	Optional	Page of the `audits` list. Default 1.
`limit`	Optional	Rows per page. Default 20, maximum 100.
`mopuan_employee_id`	Optional	Only work orders assigned to this user.
`lang`	Optional	Label language.

```
curl -X POST "https://<api-host>/v1/SRVGetAiVisorOverview" \
 -d "api_key=<api_key>" \
 -d "start_date=2026-10-07" -d "end_date=2026-11-06" \
 -d "limit=20"
```


```
{"result": {
 "result": "success",
 "filters": {"start_date": "2026-10-07", "end_date": "2026-11-06", "branch_ids": [],
   "verdict": "all", "mopuan_employee_id": null},
 "stats": {"audits": 120, "approved": 80, "failed": 15, "needs_review": 25,
   "approved_pct": 67, "failed_pct": 13, "needs_review_pct": 21, "with_issues_pct": 30,
   "cleaning_avg": 8.2, "coverage_avg": 7.9},
 "trend": [{"date": "2026-11-05", "audits": 6, "cleaning_avg": 8.4, "coverage_avg": 8.0}],
 "branches": [{"branch_id": 12, "branch_name": "Highbury", "audits": 40,
   "cleaning_avg": 8.5, "coverage_avg": 8.1, "issue_rate_pct": 20, "failed_rate_pct": 10}],
 "top_issues": [{"label": "Bed not properly made", "severity": "high", "count": 7}],
 "audits": {
   "total": 120, "page": 1, "pages": 6, "limit": 20,
   "data": [{"inspection_id": 991, "work_order_id": 12345,
     "audited_at": "2026-11-05 14:02:11", "branch_id": 12, "branch_name": "Highbury",
     "room_id": 88, "room_name": "1.17", "service_name": "Checkout clean", "cleaner_id": 45,
     "cleaner_name": "Jane Smith", "cleaner_mopuan_id": "778", "verdict": "failed",
     "review_state": "new", "cleaning_score": 6.5, "coverage_score": 8.0,
     "open_issues": 2}]
 },
 "display": {"lang": "en", "labels": {}, "verdict_tones": {}, "band_colors": {},
   "outcome_colors": {}}
}}
```


Field	Notes
`filters`	The filters used after defaults.
`stats`	Totals for the range: counts, percentages and average scores (0 to 10, `null` when no score).
`trend`	One row per day with at least one audit.

`branches`	One row per building, most audits first.
`top_issues`	Up to 8 most frequent open issues.
`audits`	Paged list: `total`, `page`, `pages`, `limit`, `data`.
`audits.data[].verdict`	`approved`, `failed`, `review_likely_fail`, `review_likely_pass` or `review_low_coverage`.
`audits.data[].cleaner_*`	The user who finished the work order, or the assigned user if none is recorded.
`display`	Labels and colours for drawing the screens.

`error_code`	Meaning
10040	AI Visor is not enabled for this company.
10015, 10016, 10099	Common errors, see section 1.2.

11.2 SRVGetAiVisorReport

The full audit report of one work order: verdict, scores, issues, checklist, gauges, room history and the two video URLs.

Field	Required	Notes
`api_key`	**Required**	See section 1.3.
`work_order_id`	**Required**	Id of the work order.
`mopuan_employee_id`	Optional	Only allow a work order assigned to this user.
`lang`	Optional	Label language.

```
curl -X POST "https://<api-host>/v1/SRVGetAiVisorReport" \
 -d "api_key=<api_key>" \
 -d "work_order_id=12345"
```

Response (shortened):

```
{"result": {
 "result": "success",
 "data": {
   "work_order": {"id": 12345, "status": 3, "service_name": "Checkout clean",
     "branch_id": 12, "branch_name": "Highbury", "room_id": 88, "room_name": "1.17",
     "cleaner_id": 45, "cleaner_name": "Jane Smith", "cleaner_mopuan_id": "778",
     "appoinment_date": "2026-11-05 10:00:00", "start_date": "2026-11-05 10:05:00",
     "end_date": "2026-11-05 11:20:00", "video_before_url": "https://.../before.mp4",
     "video_after_url": "https://.../proof.mp4"},
   "audit": {"inspection_id": 991, "verdict": "failed", "review_state": "new",
     "reviewed_at": null, "reviewed_by_name": "", "requires_review": false,
```

```
    "cleaning_score": 6.5, "coverage_score": 8.0, "checkout_dirtiness": 5.0,
    "summary": "Bed was not made after cleaning.", "analysed_at": "2026-11-05 14:02:11",
    "received_at": "2026-11-05 14:02:15", "model_version": "model / prompt-v4",
    "is_studio": false, "external_room_ref": "1.17Highbury", "room_ref_match": true,
    "videos_changed": false},
  "issues": [],
  "checklist": [],
  "checklist_outcome": {"verified_clean": 12, "visible_but_not_clean": 1,
    "not_filmed": 1},
  "gauges": [],
  "before_after_chart": {"rows": [], "skipped": 0},
  "tag_summaries": [],
  "coverage_gaps": [],
  "items_left_by_guest": [],
  "items_missing_after": [],
  "unverified_findings": [],
  "room_history": []
 },
 "display": {"lang": "en", "labels": {}}
}}
```


Field	Notes
`data.work_order`	The work order: names, cleaner, dates (converted) and the two video URLs. Note `appoinment_date`.
`data.audit`	The latest audit: verdict, review state, scores (0 to 10), summary, times (converted), model, room match and `videos_changed` (`true` when the videos were replaced after this audit).
`data.issues`	The audit's issues, with severity, fault, action, state and video timestamps.
`data.checklist`, `data.checklist_outcome`	One row per checklist item, and the count per outcome.
`data.gauges`	The score dials with the room average and the change from it.
`data.before_after_chart`	Before and after scores per item.
`data.tag_summaries`, `data.coverage_gaps`, `data.items_left_by_guest`, `data.items_missing_after`, `data.unverified_findings`	Text lists from the AI report. `[]` when empty.
`data.room_history`	Latest audit of up to 12 recent work orders of the same room, oldest first.
`display`	Labels and colours.

`error_code`	Meaning
10032	`work_order_id` is missing.

10034	The work order does not exist in this company, or is not assigned to `mopuan_employee_id`.
10040	AI Visor is not enabled for this company.
10041	This work order has no AI audit yet.
10015, 10016, 10099	Common errors, see section 1.2.

12. Recommendations

- **Send form fields, not JSON.** Parameters in a JSON body are ignored without an error, so filters and paging silently stop working.
- **Page the lists.** Use `limit` (20 to 50) and `before_id` for the work order list and the notifications. Without paging every row of the company is returned.
- **Use `reference_id` for the detail screen** instead of filtering the full list.
- **Always send `mopuan_user_id` to `SRVGetNotifications`.** Without it you get every notification of the company.
- **Never repeat a create after a `result: "1"` response**, even when `data` has only `id`. After a `10099` on a save, load the data to check whether it was saved before you retry.
- **Send `appointment_date` back unchanged** when you edit other fields. A different value counts as a reschedule and notifies the assignee.
- **Always send the required ids when you create a work order** (`customer`, `property`, `room_flat`, `service`, `appointment_date`). The service does not check them.
- **Send the author's `user_id` with every note.** Without it the note is saved with no author.
- **Send every field when you edit a customer, building or room.** Fields that are not sent are saved empty.
- **Handle `10030`.** Show the `shortfalls` to the user and let them change the quantities.
- **Hide deleted customers.** `SRVGetCustomers` lists them with `is_deleted` set to 1.
- **Never show or store the `password` field** of a customer row.
- **Do not trust calendar range edges across time zones.** `appoinment_start_date` and `appoinment_end_date` are compared with server time (Europe/London). If the company is in another time zone, ask for a slightly wider range and filter again by `appoinment_date`.
- **Show names, titles and notes as text.** Do not render them as HTML.