OpSense Mobile v2 API

Usage guide for the OpSense mobile app (v2)

opsenseapi · Version 1.1 · 6 November 2026

1. What is new

These changes support the mobile v2 screens: work order list, detail, create and edit forms, the calendar, the notifications feed, the operations dashboard and the module menu. All of them are live.

Service	Change	Screen
`SRVGetWorkOrders`	New fields `customer_name`, `service_name`. Optional paging with `limit` and `before_id`.	Work order list, detail, calendar
`SRVAddWorkOrder`	New `stock_branch_id` and `send_notification_customer`. Products sent on create are saved. The assignee is notified on create, reassign and reschedule.	New / edit work order
`SRVGetStockLocations`	New. Source location list for the work order form.	New / edit work order
`SRVGetServiceDurations`	New. Average task duration per service.	Operations dashboard
`SRVGetNotifications`	New fields `is_read`, `read_date`.	Notifications
`SRVMarkNotificationsRead`	New. Marks one notification, or all, as read.	Notifications
`SRVGetWeeklyRoute`	Date ranges that cross a month now return the right days.	Calendar, staff status, dashboard
`SRVGetUserRoles`	New field `locked_menus`.	Module menu

Existing calls keep working. Every new parameter is optional, and a response only gains fields.

2. Requests and authentication

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

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 (work order videos) are sent as `multipart/form-data`.

The `api_key` decides the company. You only ever read or change that company's data.

Every response is wrapped the same way:

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

Situation	Response
`api_key` missing	`error_code` 10015
`api_key` invalid or inactive	`error_code` 10016
The request could not be completed on our side	`error_code` 10099. Safe to retry.

Dates you send and dates you receive are in the company's time zone, `YYYY-MM-DD HH:MM:SS`, with two exceptions: the `SRVGetWorkOrders` filters `appoinment_start_date` and `appoinment_end_date` are compared with server time (Europe/London), and note dates (`notes[].added_date`) come back in server time. For a company in Europe/London there is no difference.

3. Work orders

3.1 SRVGetWorkOrders

Field	Required	Notes
`api_key`	**Required**	See section 2.
`reference_id`	Optional	Return only this work order. Use it for the detail screen.
`appoinment_start_date`	Optional	Appointments from this date and time. Note the spelling: `appoinment`. Compared with server time (Europe/London), not converted.
`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 this Mopuan employee.
`limit`	Optional, **new**	Page size. Returns the newest `limit` work orders (highest id first).
`before_id`	Optional, **new**	Only work orders with an id lower than this. Send the `next_before_id` of the previous page.

Without `limit` and `before_id` the service behaves exactly as before: every matching work order, 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` = `next_before_id`. To refresh, ask for the first page again. Work orders created while the user scrolls do not shift the pages.

For the calendar, send `appoinment_start_date` and `appoinment_end_date` for the visible month or week instead of paging.

```
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 (shortened):


```
{
 "result": {
   "result": "success",
   "data": [
     {
       "id": "3492",
       "customer_id": "55",
       "customer_name": "DMC (Londonist)",
       "service": "4",
       "service_name": "Room Cleaning",
       "branch": "12",
       "branch_name": "Highbury Residence",
       "room": "301",
       "room_name": "Gold Ensuite Standard View Mid Level (G.04)",
       "room_type_name": "Gold Ensuite",
       "appoinment_date": "2026-11-06 09:00:00",
       "start_date": "2026-11-06 09:02:00",
       "end_date": "2026-11-06 09:17:00",
       "status": "3",
       "priority": "1",
       "is_vip": "0",
       "supervised": "0",
       "assigned_to": "88",
       "assigned_name": "Nikhil Test HK",
       "assigned_photo": "u88.jpg",
       "stock_branch_id": null,
       "video": "wo/7/proof_3492.mp4",
       "prefix_video": "https://<cdn-host>/",
       "video_before": "wo/7/before_3492.mp4",
       "prefix_video_before": "https://<cdn-host>/",
       "products": [{"item_id": "41", "qty": "2", "price": "50.00",
         "package_action": null}],
       "notes": [{"id": "501", "note": "Guest checks in at 14:00",
         "added_by_name": "Jane Smith", "added_date": "2026-11-06 08:12:00"}]
     }
   ],
   "has_more": true,
   "next_before_id": 3473
 }
}
```


Field	Notes
`customer_name`	**New.** 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`	**New.** Name of the service.

`status`	1 pending, 2 in progress, 3 completed.
`priority`	1 to 4.
`stock_branch_id`	Source location of the materials. `null` means the central warehouse.
`video`, `video_before`	Add `prefix_video` / `prefix_video_before` in front to get the full URL. `null` if no video was uploaded.
`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`.

The other fields are unchanged.

3.2 SRVAddWorkOrder

Creates a work order, or edits one when `edit_id` is sent. On edit, only the fields you send change.

Field	Required	Notes
`api_key`	**Required**	See section 2.
`edit_id`	**Edit only**	Id of the work order to edit. Leave out to create.
`customer`	**Create**	Customer id (`SRVGetCustomers`).
`property`	**Create**	Building id (`SRVGetProperties`).
`room_flat`	**Create**	Room id. Rooms are listed under each building in `SRVGetProperties`.
`service`	**Create**	Service id (`SRVGetServices`).
`appointment_date`	**Create**	`YYYY-MM-DD HH:MM:SS`, company time zone.
`assigned_to`	Optional	OpSense user id of the assignee.
`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.
`supervised`	Optional	1 marks the work order as supervised.
`products`	Optional	JSON text, see below. On edit, sending it replaces the whole product list.

`stock_branch_id`	Optional, **new**	Source location for the products, from `SRVGetStockLocations`. `0` = central warehouse. Leave out to keep the current location (central on create).
`send_notification_customer`	Optional, **new**	`1` sends the customer the confirmation email when an assignee is set (on create or reassign).
`video_before`, `video_after`	Optional	Video files, `multipart/form-data`.

`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 are skipped.
`qty`	Quantity. Lines with 0 are skipped.
`action`	Packages only (`is_package` = 1 in `SRVGetProducts`): `assemble` (default), `deliver` or `release`. Same as the portal.

Products on create. Products sent on create are now saved, and their stock is reserved, exactly as on edit and in the portal. Before this change they were dropped. If stock is short, nothing is saved and you get `10030` with the short items in `shortfalls`.

Who is notified. After a successful save the API notifies the assignee by email, an OpSense notification and a Mopuan push, as the portal does:

Save	Assignee gets	Customer gets
Create with `assigned_to`	"New Work Order Assigned to You"	Confirmation email, if `send_notification_customer=1` 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

A failed email or push 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"
```

Success returns the saved work order, with `products`:

```
{"result": {"result": "1", "data": {"id": "3498", "status": "1", "assigned_to": "88",
 "stock_branch_id": null, "products": [ ... ]}}}
```

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	An id belongs to another company, or `status` is not 1, 2 or 3.
10019	`priority` is not 1 to 4.
10020	Video upload failed.
10021	`products` is not valid JSON.
10022	**New.** `stock_branch_id` is not a stock location of this company.
10030	Not enough stock. `shortfalls` lists the items that are short.

Cancelling a work order is `SRVDeleteWorkOrder` with `work_order_id`. It removes the work order and gives its reserved stock back.

3.3 SRVGetStockLocations (new)

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

Field	Required	Notes

`api_key`	**Required**	See section 2.

```
{
 "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.
`group`	`central`, `internal` (own locations) or `external` (stock kept at a client site). Use it for the section headings, as the portal does.
`customer_name`	Owner of the location. Show it before the name for `external` rows.

The central warehouse is always first. Only locations a work order can be saved with are listed.

3.4 SRVGetServiceDurations (new)

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 2.
`start_date`	Optional	`YYYY-MM-DD`, company time zone. Default: 120 days before `end_date`.
`end_date`	Optional	`YYYY-MM-DD`, company time zone. Default: today.

Work orders are selected by appointment date. Only completed work orders with a start and an end count, and only durations from 5 to 600 minutes, the same rule as the portal's AI SPV.

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
`avg_minutes`	Average duration, rounded to whole minutes.
`samples`	How many work orders the average is based on. A service with no qualifying work order is not listed.

4. Notifications

4.1 SRVGetNotifications: read state

Each notification now has two more fields. Parameters and paging are unchanged.

Field	Notes
`is_read`	**New.** `1` if the employee has read it, in the app or in the portal. `0` if not.
`read_date`	**New.** When it was read, company time zone. `null` if unread.

Read state is per employee and shared with the portal: a notification read in one shows as read in the other.

4.2 SRVMarkNotificationsRead (new)

Field	Required	Notes
`api_key`	**Required**	See section 2.
`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.

Mark one, when the employee opens a notification:

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

```
 -d "mopuan_user_id=345" \
 -d "notification_id=25510"
```

Mark all, for "Mark all as read":

```
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}}
```

`updated` is the 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.

5. SRVGetWeeklyRoute: date ranges

Parameters and response are unchanged. `start_date` and `end_date` (`YYYY-MM-DD`) are now compared as whole dates. Before this change a range that crossed a month boundary, for example 28 October to 3 November, returned no days. It now returns every day in the range.

Send both dates for the week or month on screen. Without them every route day is returned.

6. SRVGetUserRoles: locked modules

The response has one new top-level field, `locked_menus`: the modules this company has switched off, as section keys. The portal hides these modules whatever the role allows.

```
{
 "result": {
   "result": "success",
   "data": [ { "user_id": 12, "mopuan_user_id": "345", "role": { "permissions": { ... } },
     ... } ],
   "locked_menus": ["kpi-dashboard", "work-order-ai-audit"]
 }
}
```

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` from the same response: `all` sees every work order, `team` sees those whose `assigned_to` is in `visibility.user_ids`, `self` only their own.

7. 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 work order list** with `limit` (20 to 50) and `before_id`. Without paging every work order of the company is returned with its notes and products.
- **Use `reference_id` for the detail screen** instead of filtering the full list.
- **Never repeat a create after a `result: "1"` response**, even when `data` has only `id`.
- **Send `appointment_date` back unchanged** when you edit other fields. A different value counts as a reschedule and notifies the assignee.
- **Show names, titles and notes as text.** Do not render them as HTML.