AI Visor API

Read access to the AI cleaning audits shown on the portal's AI Visor page

opsenseapi · Version 1.0 · 20 October 2026

1. What these services are for

The portal's **AI Visor** page shows the AI cleaning audits of a company's work orders: totals, a score trend, a building leaderboard, the most frequent issues, and a full report per work order. These two services return the same data as JSON:

Service	Returns
`SRVGetAiVisorOverview`	Totals, daily trend, per-building figures, top issues and a paginated list of audited work orders.
`SRVGetAiVisorReport`	The full audit report for one work order.

Both services are read-only. Audits are still submitted through `SRVSubmitInspection` (see the Work Order AI Inspections guide).

Each work order counts once, by its **latest** audit. A re-audit replaces the earlier result, the same way it does in the portal. A work order therefore always shows the same verdict here as in the portal.

2. Authentication

Send the company's `api_key` as a request parameter (POST field or query string). The key decides which company the data comes from.

Situation	Response
`api_key` missing	`error_code` 10015
`api_key` invalid or inactive	`error_code` 10016
AI audit switched off for the company	`error_code` 10040

Every response is wrapped the same way:

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

Showing a user only their own audits

Both services take an optional `mopuan_employee_id`. It works exactly like the same field in `SRVGetWorkOrders`: only work orders **assigned** to that user are included. Send it whenever the signed-in user may only see their own work orders.

When it is sent:

- `SRVGetAiVisorOverview` counts only that user's audits, in every part of the response (totals, trend, buildings, top issues and the list).
- `SRVGetAiVisorReport` returns `error_code` 10034 for a work order assigned to someone else, the same as for a work order that does not exist. The report's `room_history` (and the room averages in `gauges`) include only that user's own audits of the room.

When it is not sent, the services return every audit of the company, as the portal's AI Visor page does.

3. Dates and times

All dates you send and all dates you receive are in the **company's own timezone** (see `SRVGetTimezone`). Date-times use the format `Y-m-d H:i:s`.

4. Labels, colours and language

Both services accept an optional `lang` field: `en` (default), `tr`, `el` or `vi`. An unknown value falls back to `en`. Every successful response then carries a `display` object with everything needed to draw the screens the way the portal does:

```
"display": {
 "lang": "en",
 "labels": {
   "verdict":       {"approved": "Approved", "failed": "Failed",
                     "review_likely_fail": "Needs review - likely failed", ...},
   "review_state":  {"reviewed": "Reviewed by admin", "dismissed": "Dismissed by admin"},
   "issue_state":   {"open": "...", "actioned": "Actioned", "dismissed": "Dismissed"},
   "severity":      {"high": "High", "medium": "Medium", "low": "Low"},
   "fault":         {"cleaning": "Cleaning", "maintenance": "Maintenance", "guest": "Guest"},
   "action":        {"re_clean": "Re-clean", "re_film": "Re-film",
                     "maintenance_ticket": "Maintenance ticket", "deposit_review": "Deposit review"},
   "after_status":  {"verified_clean": "Clean", "visible_but_not_clean": "Not clean",
                     "not_filmed": "Not filmed"},
   "before_filmed": {"clear": "Filmed", "partial": "Partly filmed", ...},
   "tag":           {"room_condition": "Room condition",
                     "check_in_insist": "For the next check-in", ...},
   "band":          {"good": "Good", "fair": "Fair", "poor": "Poor", "very_dirty": "Very dirty",
                     "dirty": "Dirty", "tidy": "Tidy"},
   "gauge":         {"cleaning": "Cleaning", "coverage": "Coverage",
                     "checkout_dirtiness": "Check-out dirtiness"},
   "text":          {"trend_title": "Score trend", "branches_title": "Buildings",
                     "report_issues": "Issues", ...}
 },
 "verdict_tones":  {"approved": "success", "failed": "danger", "review_likely_fail": "warning", ...},
 "band_colors":    {"good": "#0ca30c", "fair": "#fab219", "poor": "#d03b3b", "dirt": "#4a3aa7"},
 "outcome_colors": {"verified_clean": "#0ca30c", "visible_but_not_clean": "#d03b3b",
                    "not_filmed": "#fab219"}
}
```

- To show a code, look it up in its group, for example `display.labels.severity[issue.severity]`. If a code is not in the table, show it as it is.
- `labels.text` holds the screen titles, column headers, filter names and empty-state messages used by the portal. Some contain `%d` or `%1$d` / `%2$d` placeholders, filled in printf style (for example `"page_of": "Page %1$d of %2$d"`).
- The wording is the same as in the portal, in every language.

5. Verdicts

verdict	Meaning
`approved`	Nothing to re-clean, and the AI did not ask for a review.
`failed`	The AI is confident at least one item must be cleaned again.
`review_likely_fail`	An item may need cleaning again, but the AI is not confident.
`review_likely_pass`	Nothing to re-clean, but the AI asked for a human check (for example, a maintenance issue).
`review_low_coverage`	Nothing to re-clean, but the videos covered too little of the checklist (coverage below 7/10).

`review_state` is the admin's own decision on the audit: `new`, `reviewed` or `dismissed`. It does not change `verdict`.

All scores are on a 0–10 scale. For `cleaning_score` and `coverage_score`, higher is better. For `checkout_dirtiness`, higher means the room was dirtier before cleaning.

6. SRVGetAiVisorOverview

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

Field	Required	Default	Notes
`api_key`	**Required**	–	Company access key. See section 2.
`start_date`	Optional	30 days ago	`Y-m-d`. An invalid date falls back to the default.
`end_date`	Optional	today	`Y-m-d`. Swapped with `start_date` if it is earlier.
`branch_ids`	Optional	all buildings	Comma-separated building ids (`12,15`), or an array (`branch_ids[]=12&branch_ids[]=15`).
`verdict`	Optional	`all`	`all`, `approved`, `failed` or `review` (all three review verdicts). Filters the `audits` list only.
`page`	Optional	1	Page of the `audits` list. A page past the end returns the last page.
`limit`	Optional	20	Rows per page, max 100.
`mopuan_employee_id`	Optional	all users	Only work orders assigned to this user. See section 2.

`lang`	Optional	`en`	Language of `display.labels`. See section 4.

The date range and `branch_ids` apply to every part of the response. `verdict` only filters the `audits` list. The totals, trend, buildings and top issues always cover every verdict in the range. The `filters` object in the response shows the values actually used.

Response

```
{"result": {
 "result": "success",
 "filters": {"start_date": "2026-09-20", "end_date": "2026-10-20", "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": "2026-10-19", "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": "2026-10-19 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": "u-778", "verdict": "failed", "review_state": "new",
      "cleaning_score": 6.5, "coverage_score": 8.0, "open_issues": 2}
   ]
 },
 "display": { ... see section 4 ... }
}}
```

- `stats.with_issues_pct`, `branches[].issue_rate_pct` and `audits.data[].open_issues` count only open issues (not yet actioned or dismissed by an admin).
- `trend` has one row per day that has at least one audit, oldest first.
- `branches` is sorted by number of audits, highest first.
- `top_issues` lists up to 8 of the most frequent open issues.
- `audits.data` is sorted by `audited_at`, newest first.
- Averages are `null` when no audit in the group has that score.
- The cleaner is the user who finished the work order, or the assigned user if nobody is recorded as finishing it.

Example

```
curl "https://<api-host>/v1/SRVGetAiVisorOverview" \
 --data-urlencode "api_key=YOUR_API_KEY" \
 --data-urlencode "start_date=2026-10-01" \
 --data-urlencode "end_date=2026-10-20" \
```

```
 --data-urlencode "branch_ids=12,15" \
 --data-urlencode "verdict=review" \
 --data-urlencode "lang=en"
```

7. SRVGetAiVisorReport

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

Field	Required	Notes
`api_key`	**Required**	Company access key. See section 2.
`work_order_id`	**Required**	Id of the work order.
`mopuan_employee_id`	Optional	Only allow a work order assigned to this user. See section 2.
`lang`	Optional	Language of `display.labels` and `tag_summaries[].label`. Default `en`.

Errors

Situation	Response
`work_order_id` missing or not a number	`error_code` 10032
Work order not found in this company, or not assigned to `mopuan_employee_id`	`error_code` 10034
Work order has no AI audit yet	`error_code` 10041

Response


```
{"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": "u-778",
     "appoinment_date": "2026-10-19 10:00:00", "start_date": "2026-10-19 10:05:00",
     "end_date": "2026-10-19 11:20:00",
     "video_before_url": "https://.../before_12345.mp4",
     "video_after_url": "https://.../proof_12345.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": "2026-10-19 14:02:11", "received_at": "2026-10-19 14:02:15",
     "model_version": "gemini-2.x / prompt-v4", "is_studio": false,
```

```
    "external_room_ref": "1.17Highbury", "room_ref_match": true, "videos_changed": false
  },
  "issues": [
    {"checklist_key": "bed_made", "title": "Bed not properly made", "detail": "...",
     "severity": "high", "fault": "cleaning", "action": "re_clean", "state": "open",
     "before_time_sec": 12, "after_time_sec": 42}
  ],
  "checklist": [
    {"key": "bed_made", "label": "Bed made", "needs_attention": true,
     "verdict": "Bed was filmed but the duvet is rolled and the mattress is exposed.",
     "action": "re_clean",
     "before_filmed": "clear", "before_dirtiness": 6, "before_note": "...", "before_time_sec": 12,
     "after_status": "visible_but_not_clean", "after_cleanliness": 4, "after_note": "...",
     "after_time_sec": 42}
  ],
  "checklist_outcome": {"verified_clean": 12, "visible_but_not_clean": 1, "not_filmed": 1},
  "gauges": [
    {"key": "cleaning", "value": 6.5, "band": "fair", "band_key": "fair",
     "room_avg": 8.5, "delta": -2.0},
    {"key": "coverage", "value": 8.0, "band": "good", "band_key": "good",
     "room_avg": 8.0, "delta": 0.0},
    {"key": "checkout_dirtiness", "value": 5.0, "band": "dirt", "band_key": "dirty",
     "room_avg": null, "delta": null}
  ],
  "before_after_chart": {
    "rows": [{"key": "bed_made", "label": "Bed made", "before": 4.0, "after": 4.0}],
    "skipped": 1
  },
  "tag_summaries": [{"key": "check_in_insist", "label": "For the next check-in", "text": "..."}],
  "coverage_gaps": ["Wardrobe interior"],
  "items_left_by_guest": ["Bottle - left on the kitchen counter"],
  "items_missing_after": [],
  "unverified_findings": [],
  "room_history": [
    {"work_order_id": 12001, "audited_at": "2026-10-05 13:10:00",
     "cleaning_score": 8.5, "coverage_score": 8.0, "is_current": false},
    {"work_order_id": 12345, "audited_at": "2026-10-19 14:02:11",
     "cleaning_score": 6.5, "coverage_score": 8.0, "is_current": true}
  ]
 },
 "display": { ... see section 4 ... }
}}
```


Note: text written by the AI (`summary`, `issues[].title`, `checklist[].label` and so on) comes back in the report's own language whatever `lang` is. `lang` only changes the `display` labels and `tag_summaries[].label`.

Field notes

- **`issues`** — sorted by severity, highest first. `state` is the admin's action on the issue: `open`, `actioned` or `dismissed`. `action` is one of `none`, `re_clean`, `re_film`, `maintenance_ticket` or `deposit_review`. `fault` is `cleaning`, `maintenance` or `guest`.
- **`before_time_sec`** / **`after_time_sec`** — the second in `video_before_url` / `video_after_url` where the item or issue can be seen. `null` when the AI gave no timestamp.
- **`checklist`** — one row per checklist item. Items with `needs_attention: true` come first. `verdict` is the AI's one-sentence note on the item. `after_status` is `verified_clean`, `visible_but_not_clean` or `not_filmed`. `before_filmed` is `clear`, `partial`, `not_filmed` or `not_present_or_not_filmed`.
- **`checklist_outcome`** — number of checklist items per `after_status`; colours are in `display.outcome_colors`.

- **`gauges`** — the score dials. `band` picks the colour from `display.band_colors`. `band_key` picks the text from `display.labels.band`. `delta` is the change from the room's average over its other recent audits (`room_avg`). It is `null` when the room has no other audits. The check-out dirtiness gauge has no average. A gauge is left out when its score is missing.
- **`before_after_chart`** — one row per checklist item scored in both videos, on a 0–10 scale where higher is cleaner. The before score is `10 - before_dirtiness`, so both sides point the same way. `skipped` is the number of items left out because one side has no score (`display.labels.text.report_chart_skipped`).
- **`videos_changed`** — `true` when the work order's videos were replaced after this audit. The report then describes the old videos until a new audit arrives.
- **`room_ref_match`** — `false` when the room the AI reported does not match the work order's room. The audit is still stored against the work order.
- **`room_history`** — the latest audit of up to 12 recent work orders for the same room, oldest first. The audit in this report has `is_current: true`.
- Any part of the AI report can be missing. Empty lists come back as `[]` and missing values as `null` or `""`.

Example

```
curl "https://<api-host>/v1/SRVGetAiVisorReport" \
 --data-urlencode "api_key=YOUR_API_KEY" \
 --data-urlencode "work_order_id=12345" \
 --data-urlencode "lang=en"
```

8. Error codes

error_code	Meaning
10015	`api_key` missing.
10016	`api_key` invalid or inactive.
10032	`work_order_id` missing (report only).
10034	Work order not found (report only).
10040	AI Visor is not enabled for this company.
10041	The work order has no AI audit yet (report only).