USER MANUAL (IN-APP, COMPLETE)

REST API

Online: mergenhub.envest.com.tr/docs/manual-rest-api

ON THIS PAGE

General & Authentication
Main Endpoints
Examples

General & Authentication

Configuration API compatible with industry standards. Base URL: `http://<host>:8090/config/v1/project`.

Authentication	HTTP **Basic** (user/password) or the UI session cookie.
Format	JSON.
Permission	Every endpoint requires a permission; the read and write permissions are separate per module (see Roles & Permissions). A request without the permission returns 403 and the message carries the name of the required permission (`Requires 'MeteringManage, WaterManage' permission.`). If the permission is there but that action is turned off for the user, the 403 body gives the key of the action in the `action` field: `{ "code": 403, "action": "alarms.ack", "message": "Requires 'alarms.ack' permission." }` . All the keys: Action reference.
Status codes	201 (all succeeded), 207 (partial), 400 (all failed), 401 (no or invalid credentials), 403 (no permission), 409 (conflict; POST with an existing id), 415 (content type not supported), 429 (too many requests or failed sign-in lockout).

```
curl -u admin:PAROLA http://localhost:8090/config/v1/project/channels
```

WHAT CHANGED IN 3.10.1 (FOR SCRIPT WRITERS)

Permission names	`GET /auth/v1/me` gives the permissions as a list of names (`permissions`). `GET /auth/v1/capabilities` returns all the names, the role defaults, the presets and the layout of the permission window (`groups`). New names: `AlarmsManage` , `DataLoggingManage` , `Metering` , `MeteringManage` , `Water` , `WaterManage` , `Reports` , `ReportsManage` , `Workflows` , `WorkflowsManage` , `DataBridge` .
Action catalog	The `GET /auth/v1/capabilities` response now also has `actions` : `[{ key, cap, module, kind, ui }]` , 95 items. `cap` is the parent permission of the action, `module` the module key in the permission window, `kind` : create | edit | delete | run | export | manage. `ui: true` says that the action is applied only in the interface (`alarms.export` , `metering.export` , `water.export` : they have no separate endpoint on the server).
Actions turned off	The `GET /auth/v1/me` response now also has `denied` : the keys of the actions that are **effectively** turned off for the caller (after the union of their own profile and their group; empty for Root and the Administrator role). A client can look at this list before showing a button; old clients that ignore an unknown field are not affected.

Saving restrictions (Root only)	The body of `POST /auth/v1/users/{name}/permissions` is `{ permissions, useRoleDefault, restrict }`; the body of `PUT /auth/v1/groups/{name}` is `{ permissions, restrict }`. `restrict` is the array of the action keys that are turned off: if the field is not in the body the stored restrictions do not change (an old script does not delete restrictions), an empty array removes the restrictions, an unknown key returns 400 (the keys in the `unknown` field) and nothing is saved. Adding a new action to the list of a grouped user without their own permissions returns 400 (`reason: "groupProfile"`): the restriction is given to the group.
User and group records	`GET /auth/v1/users` : `permissions` is the effective set (own ∪ group); `ownPermissions` is the user's own set (`null` if they have no custom permissions); `restrict` and `ownRestrict` are the user's own list of turned-off actions; `denied` is the effective result; `restrictApplies` says whether their own list enters the effective result (`false` for the Administrator role and for a grouped user without their own permissions). `GET /auth/v1/groups` : `restrict` on every group.
Reading endpoints that depend on an action	Reading endpoints do not check an action; three exceptions: `GET /config/v1/logger/export` (`log.export`), `GET /config/v1/meteringreports/{id}/template` (`report.download`), `GET /uaserver/v1/instance-certificate` (`cert.instance`). Reading a report with a data access key (`mhkey_`) needs the owner's `report.download` action too.
Settings	`GET /config/v1/settings` returns filled only the sections you can change; the other sections come with default values and their names are in the `hiddenSections` list. An attempt to change a section you cannot see returns 403; on save the stored value of those sections is kept. A section whose action is turned off (e.g. with `settings.opcua` off, `opcUa`) goes into the `hiddenSections` list as well; the 403 body of an attempt to change such a section carries the section names (`denied`) and the action (`action`). The `GET /ai/v1/settings` response has the `canProvider` , `canPrompt` , `canSkills` flags.
License, health	`GET /config/v1/license` gives only the status without the Server settings permission (`limited: true`). `GET /health` returns only `product` , `status` , `tagCount` to a remote request without sign-in.
Creating records	`POST /config/v1/scada/screens` and `POST /config/v1/logger/groups` return 409 if the body carries the id of an existing record; use PUT to update. `POST /config/v1/project/import/bundle` accepts only `Content-Type: application/json` (415).
Basic sign-in lockout	8 wrong passwords in 5 minutes: a 5-minute lockout per username (429).
Behind a proxy	`X-Forwarded-*` headers are accepted only from a trusted proxy.

Main Endpoints

GET	`/config/v1/project` — project info
GET/POST	`/config/v1/project/channels` — channels
GET/POST	`…/channels/{ch}/devices` — devices
PUT	…/devices/{dev} — update device
POST	…/devices/{dev}/tags — add tag (single/array=batch)
GET	`/runtime/v1/status` — counters
GET	`/runtime/v1/values?prefix=Kanal.Cihaz` — live value
POST	`/runtime/v1/write` — `{"path":"K.C.Tag","value":1}`

Examples

```
# Kanaldaki cihazlar
curl -u admin:PAROLA http://localhost:8090/config/v1/project/channels/Modbus_1/devices

# Değer yaz
curl -u admin:PAROLA -X POST http://localhost:8090/runtime/v1/write \
  -H "Content-Type: application/json" -d '{"path":"Modbus_1.Dev1.Setpoint","value":42}'

# Canlı değer oku
curl -u admin:PAROLA "http://localhost:8090/runtime/v1/values?prefix=Modbus_1.Dev1"
```