



Sigur Public REST API Developer's Guide

Revision dated 23.09.2026.

Table of Contents

1.	Overview	6
2.	Common REST API	7
2.1.	Authentication	7
2.1.1.	Authenticate user	7
2.1.2.	Refresh JWT access token	8
2.1.3.	Deprecated: Authenticate user	9
2.2.	Departments	10
2.2.1.	Get all departments	10
2.2.2.	Get department by ID	12
2.2.3.	Count departments	13
2.2.4.	Create department	14
2.2.5.	Update department	15
2.2.6.	Delete department	16
2.2.7.	Create multiple departments	17
2.2.8.	Update multiple departments	19
2.2.9.	Delete multiple departments	21
2.3.	Positions	22
2.3.1.	Get all positions	22
2.3.2.	Get position by ID	24
2.3.3.	Create position	25
2.3.4.	Update position	26
2.3.5.	Delete position	27
2.3.6.	Create multiple positions	28
2.3.7.	Update multiple positions	30
2.3.8.	Delete multiple positions	32
2.4.	Cards	33
2.4.1.	Overview	33
2.4.2.	Get all cards	34
2.4.3.	Get card by ID	36
2.4.4.	Create card	37
2.4.5.	Update card	38
2.4.6.	Delete card	39
2.4.7.	Create multiple cards	40
2.4.8.	Update multiple cards	42
2.4.9.	Delete multiple cards	44
2.5.	Employees	45
2.5.1.	Get all employees	45
2.5.2.	Get employee by ID	48
2.5.3.	Count employees	51
2.5.4.	Create employee	53
2.5.5.	Update employee	56
2.5.6.	Delete employee	59
2.5.7.	Block employee	61
2.5.8.	Unblock employee	62
2.5.9.	Create multiple employees	63
2.5.10.	Update multiple employees	66

2.5.11.	Delete multiple Employees	69
2.5.12.	Block multiple employees	70
2.5.13.	Unblock multiple employees	71
2.6.	Official vehicles	72
2.6.1.	Get all official vehicles	72
2.6.2.	Get official vehicle by ID	74
2.6.3.	Create official vehicle	75
2.6.4.	Update official vehicle	76
2.6.5.	Delete official vehicle	77
2.6.6.	Create multiple official vehicles	78
2.6.7.	Update multiple official vehicles	80
2.6.8.	Delete multiple official vehicles	82
2.7.	Deprecated: Employee vehicles	83
2.7.1.	Add new employee vehicle	83
2.7.2.	Update employee vehicle	84
2.7.3.	Delete employee vehicle	85
2.8.	Bindings	86
2.8.1.	Employees-Cards	86
2.8.1.1.	Get Employee - Card bindings	86
2.8.1.2.	Assign Cards to Employees	87
2.8.1.3.	Update expiration time for the specified Cards assigned to Employees	88
2.8.1.4.	Update start and expiration time for the specified Cards assigned to Employees	89
2.8.1.5.	Remove Cards from Employees	90
2.8.2.	Employees-Guest Cards	91
2.8.2.1.	Get Employee-Guest Card bindings	91
2.8.2.2.	Assign Guest Cards to Employees	93
2.8.2.3.	Update Guest Cards assigned to Employees	94
2.8.2.4.	Withdraw Guest Cards assigned to Employees	95
2.8.3.	Employees-Access Rules	96
2.8.3.1.	Get Employees Access Rules	96
2.8.3.2.	Assign Access Rules to Employees	97
2.8.3.3.	Remove Access Rules from Employees	98
2.8.4.	Employees-Access Points	99
2.8.4.1.	Get bindings between Employees and Access Points	99
2.8.4.2.	Create bindings between Employees and Access Points	100
2.8.4.3.	Delete bindings between Employees and Access Points	101
2.8.5.	Official Vehicles-Cards	102
2.8.5.1.	Get Official Vehicles Cards	102
2.8.5.2.	Update expiration time for Cards of Official Vehicles	103
2.8.5.3.	Assign Cards to Official Vehicles	104
2.8.5.4.	Remove Cards from Official Vehicles	105
2.8.6.	Official Vehicles-Access Rules	106
2.8.6.1.	Get Official Vehicles Access Rules	106
2.8.6.2.	Assign Access Rules to Official Vehicle	107

2.8.6.3.	Remove Access Rules from Official Vehicles	108
2.8.7.	Official Vehicles-Access Points	109
2.8.7.1.	Get bindings between Official Vehicles and Access Points	109
2.8.7.2.	Create bindings between Vehicles and Access Points	110
2.8.7.3.	Delete bindings between Vehicles and Access Points	111
2.8.8.	Access Rules-Access Points	112
2.8.8.1.	Get bindings between Access Rules and Access Points	112
2.8.8.2.	Assign Access Aoints to Access Rules	113
2.8.8.3.	Unassign Access Points to Access Rules	114
2.8.9.	Guests-Access Rules	115
2.8.9.1.	Get Guests Access Rules	115
2.8.9.2.	Assign Access Rules to Guests	116
2.8.9.3.	Unassign Access Rules from Guests	117
2.9.	Access rules	118
2.9.1.	Get all access rules	118
2.9.2.	Get access rule by ID	121
2.9.3.	Create access rule	123
2.9.4.	Update access rule	126
2.9.5.	Delete access rule	129
2.9.6.	Count access rules	130
2.9.7.	Get all folders of access rules	132
2.10.	Access points	133
2.10.1.	Get all access points	133
2.10.2.	Get access point by ID	135
2.10.3.	Get all folders of access points	136
2.10.4.	Count access points	137
2.11.	Users	138
2.11.1.	Get all users	138
2.11.2.	Get user by ID	139
2.11.3.	Create user from an existing employee	140
2.11.4.	Update user	141
2.11.5.	Delete user	142
2.12.	Zones	143
2.12.1.	Get all zones	143
2.12.2.	Get zone by ID	145
2.13.	Custom fields	146
2.13.1.	Overview	146
2.13.2.	Get all custom fields	147
2.13.3.	Create custom field	149
2.13.4.	Update custom field	150
2.13.5.	Delete custom field	151
2.14.	Events	152
2.14.1.	Get all event types	152
2.14.2.	Get list of events	153
2.14.3.	Get all parsed events	155

2.15.	Guests	173
2.15.1.	Get guests	173
2.15.2.	Create guests with cards	175
2.15.3.	Update guests	177
2.15.4.	Remove guests	179
2.15.5.	Get guest document types	180
2.15.6.	Get guest full photo by ID	181
2.16.	Guest cards	182
2.16.1.	Get all guest cards	182
2.16.2.	Get guest card by ID	184
2.16.3.	Create guest card	185
2.16.4.	Update guest card	187
2.16.5.	Delete guest card	189
2.16.6.	Create multiple guest cards	190
2.16.7.	Update multiple guest cards	192
2.16.8.	Delete multiple guest cards	194
2.17.	Guest QR codes	195
2.17.1.	Get guest QR code by ID	195
2.17.2.	Create guests with QR codes	196
2.17.3.	Update guest QR code	198
2.18.	Employee virtual identifiers (mobile ID, PIN, QR code)	199
2.18.1.	Overview	199
2.18.2.	Assign virtual identifier to employee	200
2.18.3.	Update start and expiration time for virtual identifiers	203
2.18.4.	Remove virtual identifier	205
2.18.5.	Generate link for mobile app	206
2.19.	Roles	207
2.19.1.	Get all roles	207
2.19.2.	Get role by ID	209
2.19.3.	Assign role to user	210
2.19.4.	Unassign role from user	211
3.	Response error codes	212
3.1.	400 Invalid Request	212
3.2.	401 Unauthorized	212
3.3.	403 Forbidden	212
3.4.	404 Not Found	212
3.5.	422 Unprocessable Entity	212
3.6.	500 Internal Server Error	213
3.7.	503 Service Unavailable	213
4.	Contacts	214

1. Overview

This guide describes the public REST API of Sigur Access Control and Management System (ACS), including request/response descriptions and examples. This guide applies to software version 1.8.5.69.

The REST APIs are available at: <http://<Sigur server IP address>:9500/api/v1/<endpoint>>.

Swagger (OAS) specifications are available at: <http://<Sigur server IP address>:9500/swagger>.

Wherever *<custom field>:<field type>* appears, custom fields are used.

All GET requests return 50 records by default, with a maximum of 3000 records, except for the GET `api/v1/employees` request, which has no limit set. To retrieve a large number of records, combine the *limit* and *offset* parameters and page through the results.

Filtering parameters may include comparison operators in the form *<field>[operation]=<value>*. Unless otherwise specified, the following operations are commonly used:

Operator	Description
IN	Matches one of the provided values (comma-separated list)
EQ	Equals the value
NEQ	Not equal to value
GT	Greater than value
LT	Less than value
GTE	Greater than or equal to value
LTE	Less than or equal to value
IS_NULL	The field is null
IS_NOT_NULL	The field is not null
STARTS_WITH	Starts with value
CONTAINS	Contains value

2. Common REST API

2.1. Authentication

2.1.1. Authenticate user

A session key (JWT token) is required to access the Public REST API. Include the key in the Authorization header. The key is valid for one hour; after it expires, generate a new one.

URL Endpoint	/api/v1/users/auth
Method	POST
Header Parameters	Content-Type: application/json
Path Parameters	
Query Parameters	
Body	<pre>{ "username": "string", "password": "string" }</pre> Required fields: - username — system user's login - password — system user's password
Success Response	Status code: 200 Content-type: application/json Body: <pre>{ "token": "string", "refreshToken": "string", "expiresAt": "2023-12-03T13:29:50.590Z", "refreshExpiresAt": "2023-12-03T13:29:50.590Z" }</pre>
Error Response	401 (Unauthorized) 403 (Forbidden) 500 (Internal Server Error)
Example	Request: <pre>curl -X 'POST' \ 'http://localhost:9500/api/v1/users/auth' \ -H 'Content-Type: application/json' \ -d '{ "username": "username", "password": "123456789" }'</pre> Response: <pre>{ "token": "eyJhbGciOiJSUz...", "refreshToken": "eyJhbGciOiJSUz...", "expiresAt": "2023-12-03T17:38:59.931748+03:00", "refreshExpiresAt": "2023-12-04T16:39:00.1471714+03:00" }</pre>

2.1.2. Refresh JWT access token

URL Endpoint	/api/v1/jwt/refresh
Method	POST
Header Parameters	Content-Type: application/json
Path Parameters	
Query Parameters	
Body	<String>
Success Response	Status code: 200 Content-type: application/json Body: <pre>{ "token": "string", "refreshToken": "string", "expiresAt": "2023-12-03T18:08:10.737Z", "refreshExpiresAt": "2023-12-03T18:08:10.737Z" }</pre>
Error Response	400 (Invalid Request) 500 (Internal Server Error)
Example	Request: <pre>curl -X 'POST' \ 'http://localhost:9500/api/v1/jwt/refresh' \ -H 'Content-Type: application/json' \ -d 'eyJhbGciOiJSUzI1NiJ9.eyJzdWIiOiJ1c2Vyb' Response:</pre> <pre>{ "token": "yaQ0UyicRoYd5A60DhT08vTW-KOC9_NvhLMG", "refreshToken": "eyJhbGciOiJSUzI1NiJ9.eyJzdWIiOiJ1c2Vyb", "expiresAt": "2023-12-03T17:38:59.931748+03:00", "refreshExpiresAt": "2023-12-04T16:39:00.1471714+03:00" }</pre>

2.1.3. Deprecated: Authenticate user

This endpoint is deprecated. Use [POST /api/v1/users/auth](#) instead.

URL Endpoint	api/v1/application-keys/auth
Method	POST
Header Parameters	Content-Type: application/json
Path Parameters	
Query Parameters	
Body	<pre>{ "system": <String>, "password": <String> }</pre> Required fields: - system — system user's login - password — system user's password
Success Response	Status code: 200 Content-type: application/json Body: <Session key>
Error Response	400 (Invalid Request) 401 (Unauthorized) 500 (Internal Server Error)
Example	Request: <pre>curl -X POST http://localhost:9500/api/v1/application-keys/auth -H "Content-Type: application/json" -d 'ahe5aNJ...'</pre> Response: <pre>eyJraWQiOiJh...</pre>

2.2. Departments

2.2.1. Get all departments

URL Endpoint	api/v1/departments
Method	GET
Header Parameters	Authorization: Bearer <JWT access token>
Path Parameters	
Query Parameters	id – identifier(s) of Department(s). Returns only records with the specified identifiers (e.g. api/v1/departments?id=1,2 or api/v1/departments? id=1&id=2) parentId – identifier of a parent department. If you want to get only root departments, then use parentId=0 (e.g. api/v1/departments?parentId=1) name – filter by department name (e.g. api/v1/departments?name=Warehouse) limit – specifies how many entities the response should contain (e.g. api/v1/departments?limit=10 – means that only 10 entities returned in a response). The value of this parameter must be between 0 and 3000 offset – specifies how many entities should be skipped from the top (e.g. api/v1/departments?limit=10&offset=10 – means that 10 records are skipped, and next 10 entities are returned). The value of this parameter must be greater than or equal to 0 <any custom field> – e.g. custom field "externalId" has been created (e.g. api/v1/departments?externalId=135) includeFields – specifies what fields the response should contain (e.g. api/v1/departments?includeFields=name,parentId – only fields "name" and "parentId" are returned) excludeFields – specifies what fields should be omitted from the response (e.g. api/v1/departments?excludeFields=name – all fields except "name" are returned) sortBy – specifies sort field (e.g. api/v1/departments?sortBy=name – means that entities would be sorted by "name" field) sortOrder – specifies sort order. Can be either ASC or DESC (e.g. api/v1/departments?sortBy=name&sortOrder=DESC – means that entities would be sorted by "name" field in descending order) isGlobal – specifies whether to return only global or only local departments. All departments are returned if not specified (default) (e.g. api/v1/departments?isGlobal=true)
Body	
Success Response	Status code: 200 Content-type: application/json Body: <pre>[{ "id": <Number>, "parentId": <Number>, // ID of the parent department "name": <String>, "hasChildren": <Boolean>, // indicates whether this department has nested departments "description": <String>, "isGlobal": <Boolean>, // indicates whether the department is global or local "<custom field>": <Field type>, ... }, ...]</pre>
Error Response	400 (Invalid Request) 401 (Unauthorized) 403 (Access Denied) 422 (Validation Error) 500 (Internal Server Error)

Example	Request: <pre>curl -X GET http://192.168.0.108:9500/api/v1/departments -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilsm"</pre> Response: <pre>[{ "id":1, "parentid": 0, "name":"Head Department", "hasChildren": true, "description": "Main", "isGlobal": false, "externalId": 135 }, { "id":2, "parentid":1, "name":"R&D", "hasChildren": false, "isGlobal": false, "externalId": 136 }]</pre>
---------	---

2.2.2. Get department by ID

Returns a department with a specified ID according to the user's permissions.

URL Endpoint	api/v1/departments/{id}
Method	GET
Header Parameters	Authorization: Bearer <JWT access token>
Path Parameters	id – identifier of a Department
Query Parameters	includeFields – specifies what fields the response should contain (e.g. api/v1/departments/1?includeFields=name,parentId – only fields "name" and "parentId" are returned) excludeFields – specifies what fields the response should not contain (e.g. api/v1/departments/1?excludeFields=name – all fields except "name" are returned)
Body	
Success Response	Status code: 200 Content-type: application/json Body: <pre>{ "id": <Number>, "parentId": <Number>, // ID of the parent department "name": <String>, "hasChildren": <Boolean>, // indicates whether this department has nested departments "description": <String>, "isGlobal": <Boolean>, // indicates whether the department is global or local "custom field": <Field type>, ... }</pre>
Error Response	<u>400 (Invalid Request)</u> <u>401 (Unauthorized)</u> <u>403 (Access denied)</u> <u>404 (Department with specified id not found)</u> <u>500 (Internal Server Error)</u>
Example	Request: <pre>curl -X GET http://192.168.0.108:9500 /api/v1/departments/2 -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilslm"</pre> Response: <pre>{ "id": 2, "parentId": 1, "name": "R&D", "hasChildren": false, "isGlobal": false }</pre>

2.2.3. Count departments

URL Endpoint	api/v1/departments/count
Method	GET
Header Parameters	Authorization: Bearer <JWT access token>
Path Parameters	
Query Parameters	id - identifier(s) of Department(s) - returns only records with specified identifiers (e.g. api/v1/departments/count?id=1,2 or api/v1/departments/count? id=1&id=2) parentId - identifier of parent department. If you want to get only root departments then use parentId=0 (e.g. api/v1/departments/count?parentId=1) name - filter by department name (e.g. api/v1/departments/count?name=Warehouse) isRecursive - whether to count departments recursively or not (e.g. api/v1/departments/count?isRecursive=true)
Body	
Success Response	Status code: 200 Content-type: application/json Body: <pre>[<Number>]</pre>
Error Response	<u>400 (Invalid request)</u> <u>401 (Unauthorized)</u> <u>403 (Access denied)</u> <u>422 (Validation Error)</u> <u>500 (Internal Server Error)</u>
Example	Request: <pre>curl -X GET http://192.168.0.108:9500/api/v1/departments/count -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilslm"</pre> Response: <pre>[8]</pre>

2.2.4. Create department

URL Endpoint	api/v1/departments
Method	POST
Header Parameters	Authorization: Bearer <JWT access token> Content-Type: application/json
Path Parameters	
Query Parameters	
Body	<pre>{ "parentId": <Number>, "name": <String>, "description": <String>, "<custom field>": <Field type> ... }</pre> Required fields: name If "parentId" is not specified then high level department is assigned (root department).
Success Response	Status code: 200 Content-type: application/json Body: <pre>{ "id": <Number>, "parentId": <Number>, "name": <String>, "description": <String>, "isGlobal": <Boolean>, "<custom field>": <Field type> ... }</pre>
Error Response	400 (Invalid Request) 401 (Unauthorized) 403 (Access denied) 404 (Parent department is not found) 422 (Validation Error) 500 (Internal Server Error)
Example	Request: <pre>curl -X POST http://192.168.0.108:9500/api/v1/departments -H "Content-Type: application/json" -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilsm" -d '{"name":"Test department ","parentId":1, "externalId ":135}'</pre> Response: <pre>{ "id":3, "parentId":1, "name":"Test department", "isGlobal": false, "externalId":135 }</pre>

2.2.5. Update department

URL Endpoint	api/v1/departments/{id}
Method	PUT
Header Parameters	Authorization: Bearer <JWT access token> Content-Type: application/json
Path Parameters	id – identifier of Department to be edited
Query Parameters	
Body	<pre>{ "parentId": <Number>, "name": <String>, "description": <String>, "<custom field>": <Field type> ... }</pre> Body contains only fields that should be modified.
Success Response	Status code: 200 Content-type: application/json Body: <pre>{ "id": <Number>, "parentId": <Number>, "name": <String>, "hasChildren": <Boolean>, "description": <String>, "isGlobal": <Boolean>, "<custom field>": <Field type> ... }</pre>
Error Response	400 (Invalid Request) 401 (Unauthorized) 403 (Permission denied) 404 (Department with specified ID not found) 422 (Validation Error) 500 (Internal Server Error)
Example	Request: <pre>curl -X PUT http://192.168.0.108:9500/api/v1/departments/3 -H "Content-Type: application/json" -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilsm" -d {"name": "Test updated department", "parentId": 1}</pre> Response: <pre>{ "id": 3, "parentId": 1, "name": "Test updated department", "hasChildren": false, "isGlobal": false }</pre>

2.2.6. Delete department

Deletes a department with the specified ID and returns the deleted entity.

A department cannot be deleted if:

- it contains assigned employees or vehicles (recursively);
- it has nested departments (subdepartments).

All assigned entities must be unassigned before deletion.

URL Endpoint	api/v1/departments/{id}
Method	DELETE
Header Parameters	Authorization: Bearer <JWT access token> Content-Type: application/json
Path Parameters	• id - identifier of Department to be deleted
Query Parameters	
Body	
Success Response	Status code: 200 Content-type: application/json Body: <pre>{ "id": <Number>, "parentId": <Number>, "name": <String>, "hasChildren": <Boolean>, "description": <String>, "isGlobal": <Boolean>, "<custom field>": <Field type> ... }</pre>
Error Response	<u>400 (Invalid Request)</u> <u>401 (Unauthorized)</u> <u>403 (Permission denied)</u> <u>404 (Department with specified id not found)</u> <u>422 (Validation Error)</u> <u>500 (Internal Server Error)</u>
Example	Request: <pre>curl -X DELETE http://192.168.0.108:9500/api/v1/departments/3 -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilslm"</pre> Response: <pre>{ "id": 3, "parentId": 1, "name": "Test updated department", "isGlobal": false }</pre>

2.2.7. Create multiple departments

URL Endpoint	api/v1/departments/create
Method	POST
Header Parameters	Authorization: Bearer <JWT access token> Content-Type: application/json
Path Parameters	
Query Parameters	
Body	<pre>[{ "parentId": <Number>, "name": <String>, "description": <String>, "<custom field>": <Field type> ... }, { "parentId": <Number>, "name": <String>, "description": <String>, "<custom field>": <Field type> ... }, ...]</pre> Required fields: name If "parentId" is not specified then high level department is assigned (root department).
Success Response	Status code: 200 Content-type: application/json Body: <pre>[{ "id": <Number>, "name": <String>, "parentId": <Number>, "description": <String>, "isGlobal": <Boolean>, "<custom field>": <Field type>, ... }, { "id": <Number>, "name": <String>, "parentId": <Number>, "description": <String>, "isGlobal": <Boolean>, "<custom field>": <Field type>, ... }, ...]</pre>
Error Response	400 (Invalid Request) 401 (Unauthorized) 403 (Permission denied) 404 (not found) 422 (Validation Error) 500 (Internal Server Error)

Example	Request: <pre>curl -X POST http://192.168.0.108:9500/api/v1/departments/create -H "Content-Type: application/json" -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilsm" -d '[{ "name": "Test department ", "parentId": 1, "externalId ":135}, { "name": "Test department 1", "parentId": 2, "externalId ":136}]'</pre> Response: <pre>[{ "id": 3, "parentId": 1, "name": "Test department", "isGlobal": false, "externalId": 135 }, { "id": 4, "parentId": 2, "name": "Test department 1", "isGlobal": false, "externalId ": 136 }]</pre>
---------	--

2.2.8. Update multiple departments

URL Endpoint	api/v1/departments/update
Method	POST
Header Parameters	Authorization: Bearer <JWT access token> Content-Type: application/json
Path Parameters	
Query Parameters	
Body	<pre>[{ "id": <Number>, "parentId": <Number>, "name": <String>, "description": <String>, "<custom field>": <Field type> ... }, { "id": <Number>, "parentId": <Number>, "name": <String>, "description": <String>, "<custom field>": <Field type> ... }, ...]</pre> Required fields: id Body contains only fields that should be modified.
Success Response	Status code: 200 Content-type: application/json Body: <pre>[{ "id": <Number>, "parentId": <Number>, "name": <String>, "hasChildren": <boolean>, "description": <String>, "isGlobal": <Boolean>, "<custom field>": <Field type> ... }, { "id": <Number>, "parentId": <Number>, "name": <String>, "hasChildren": <boolean>, "description": <String>, "isGlobal": <Boolean>, "<custom field>": <Field type> ... }, ...]</pre>
Error Response	400 (Invalid Request) 401 (Unauthorized) 403 (Permission denied) 404 (Not Found) 422 (Validation Error) 500 (Internal Server Error)

Example	Request: <pre>curl -X POST http://192.168.0.108:9500/api/v1/departments/update -H "Content-Type: application/json" -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilsm" -d '[{ "id":3, "name":"Test department updated"}, {"id":4, "name":"Test department 1 updated"}]'</pre> Response: <pre>[{ "id": 3, "parentId": 1, "name": "Test department updated", "hasChildren": false, "description": "desc1", "isGlobal": false }, { "id": 4, "parentId": 2, "name": "Test department 1 updated", "hasChildren": false, "description": "desc2", "isGlobal": false }]</pre>
---------	---

2.2.9. Delete multiple departments

A department cannot be deleted if it contains assigned employees or vehicles (recursively). Therefore, all assigned entities must first be unassigned.

URL Endpoint	api/v1/departments/delete
Method	POST
Header Parameters	Authorization: Bearer <JWT access token> Content-Type: application/json
Path Parameters	
Query Parameters	
Body	[<Number>, <Number>] Body contains ids of departments that should be deleted.
Success Response	Status code: 200 Content-type: application/json Body: [<Number>, <Number>] Body contains ids of deleted departments.
Error Response	<u>400 (Invalid Request)</u> <u>401 (Unauthorized)</u> <u>403 (Permission denied)</u> <u>404 (Not Found)</u> <u>422 (Validation Error)</u> <u>500 (Internal Server Error)</u>
Example	Request: curl -X POST http://192.168.0.108:9500/api/v1/departments/delete -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1Y1lljoiVXNlciIsIm" -d '[3, 4]' Response: [3, 4]

2.3. Positions

2.3.1. Get all positions

URL Endpoint	api/v1/positions
Method	GET
Header Parameters	Authorization: Bearer <JWT access token>
Path Parameters	
Query Parameters	▪ id – identifier(s) of Position(s) – returns only records with specified identifiers (e.g. api/v1/positions?id=1,2 or api/v1/positions?id=1&id=2) ▪ name – filter by position name (e.g. api/v1/positions?name=Developer) ▪ limit – specifies how many entities the response should contain (e.g. api/v1/positions?limit=10 – means that only 10 entities are returned). The value of this parameter must be between 0 and 3000 ▪ offset – specifies how many entities should be skipped from the top (e.g. api/v1/positions?limit=10&offset=10 – means that 10 records are skipped and next 10 entities are returned). The value of this parameter must be greater than or equal to 0 ▪ <any custom field> – e.g. custom field "externalId" has been created (e.g. api/v1/positions?externalId=135) ▪ includeFields – specifies what fields the response should contain (e.g. api/v1/positions?includeFields=name – only the "name" field is returned) ▪ excludeFields – specifies what fields the response should not contain (e.g. api/v1/positions?excludeFields=id – all fields except "id" are returned) ▪ sortBy – specifies sort field (e.g. api/v1/positions?sortBy=name – means that entities would be sorted by the "name" field) ▪ sortOrder – specifies sort order. Can be either ASC or DESC (e.g. api/v1/positions?sortBy=name&sortOrder=DESC – means that entities would be sorted by the "name" field in descending order)
Body	
Success Response	Status code: 200 Content-type: application/json Body: <pre>[{ "id": <Number>, "name": <String>, "<custom field>": <Field type>, ... }, ...]</pre>

Error Response	400 (Invalid request) 401 (Unauthorized) 403 (Permission denied) 422 (Validation Error) 500 (Internal Server Error)
Example	Request: curl -X GET http://192.168.0.108:9500/api/v1/positions -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilslm" Response: [{ "id":1, "name":"Head of Department" }, { "id":2, "name":"Designer" }]

2.3.2. Get position by ID

URL Endpoint	api/v1/positions/{id}
Method	GET
Header Parameters	Authorization: Bearer <JWT access token>
Path Parameters	id – identifier of a Position
Query Parameters	includeFields – specifies what fields the response should contain (e.g. api/v1/positions/1?includeFields=name – only the "name" field is returned) excludeFields – specifies what fields the response should not contain (e.g. api/v1/positions/1?excludeFields=id – all fields except "id" are returned)
Body	
Success Response	Status code: 200 Content-type: application/json Body: <pre>{ "id": <Number>, "name": <String>, "<custom field>": <Field type>, ... }</pre>
Error Response	<u>400 (Invalid request)</u> <u>401 (Unauthorized)</u> <u>403 (Permission denied)</u> <u>404 (Position with specified id not found)</u> <u>500 (Internal Server Error)</u>
Example	Request: <pre>curl -X GET http://192.168.0.108:9500/api/v1/positions/2 -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilslm"</pre> Response: <pre>{ "id":2, "name":"Designer" }</pre>

2.3.3. Create position

URL Endpoint	api/v1/positions
Method	POST
Header Parameters	Authorization: Bearer <JWT access token> Content-Type: application/json
Path Parameters	
Query Parameters	
Body	<pre>{ "name": <String>, "<custom field>": <Field type>, ... }</pre> Required fields: name
Success Response	Status code: 200 Content-type: application/json Body: <pre>{ "id": <Number>, "name": <String>, "<custom field>": <Field type>, ... }</pre>
Error Response	400 (Invalid request) 401 (Unauthorized) 403 (Permission denied) 422 (Validation Error) 500 (Internal Server Error)
Example	Request: <pre>curl -X POST http://192.168.0.108:9500/api/v1/positions -H "Content-Type: application/json" -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilsm" -d '{"name": "Developer"}'</pre> Response: <pre>{ "id": 3, "name": "Developer" }</pre>

2.3.4. Update position

URL Endpoint	api/v1/positions/{id}
Method	PUT
Header Parameters	Authorization: Bearer <JWT access token> Content-Type: application/json
Path Parameters	id – identifier of Position to be edited
Query Parameters	
Body	<pre>{ "name": <String>, "<custom field>": <Field type>, ... }</pre> Body contains only fields that should be modified.
Success Response	Status code: 200 Content-type: application/json Body: <pre>{ "id": <Number>, "name": <String>, "<custom field>": <Field type>, ... }</pre>
Error Response	400 (Invalid request) 401 (Unauthorized) 403 (Permission denied) 404 (Position with specified ID not found) 422 (Validation Error) 500 (Internal Server Error)
Example	Request: <pre>curl -X PUT http://192.168.0.108:9500/api/v1/positions/3 -H "Content-Type: application/json" -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilsm" -d {"name":"Team Lead "}</pre> Response: <pre>{ "id":3, "name":"Team Lead" }</pre>

2.3.5. Delete position

A position can only be deleted if it is not assigned to an employee. Therefore, it must first be unassigned.

URL Endpoint	api/v1/positions/{id}
Method	DELETE
Header Parameters	Authorization: Bearer <JWT access token>
Path Parameters	id - identifier of Position to be deleted
Query Parameters	
Body	
Success Response	Status code: 200 Content-type: application/json Body: <pre>{ "id": <Number>, "name": <String>, "<custom field>": <Field type>, ... }</pre>
Error Response	400 (Invalid request) 401 (Unauthorized) 403 (Permission denied) 404 (Position with specified id not found) 422 (Validation Error) 500 (Internal Server Error)
Example	Request: <pre>curl -X DELETE http://192.168.0.108:9500/api/v1/positions/3 -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilsm"</pre> Response: <pre>{ "id":3, "name":"Team Lead" }</pre>

2.3.6. Create multiple positions

URL Endpoint	api/v1/positions/create
Method	POST
Header Parameters	Authorization: Bearer <JWT access token> Content-Type: application/json
Path Parameters	
Query Parameters	
Body	<pre>[{ "name": <String>, "<custom field>": <Field type>, ... }, { "name": <String>, "<custom field>": <Field type>, ... } ...]</pre> Required fields: name
Success Response	Status code: 200 Content-type: application/json Body: <pre>[{ "id": <Number>, "name": <String>, "<custom field>": <Field type>, ... }, { "id": <Number>, "name": <String>, "<custom field>": <Field type>, ... } ...]</pre>
Error Response	<u>400 (Invalid request)</u> <u>401 (Unauthorized)</u> <u>403 (Permission denied)</u> <u>422 (Validation Error)</u> <u>500 (Internal Server Error)</u>

Example	Request: <pre>curl -X POST http://localhost:9500/api/v1/positions/create -H "Content-Type: application/json" -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilslm" -d '[{ "name": "HR"}, {"name": "DevOps"}]'</pre> Response: <pre>[{ "id": 4, "name": "HR" }, { "id": 5, "name": "DevOps" }]</pre>
---------	--

2.3.7. Update multiple positions

URL Endpoint	api/v1/positions/update
Method	POST
Header Parameters	Authorization: Bearer <JWT access token> Content-Type: application/json
Path Parameters	
Query Parameters	
Body	<pre>[{ "id": <Number>, "name": <String>, "<custom field>": <Field type>, ... }, { "id": <Number>, "name": <String>, "<custom field>": <Field type>, ... } ...]</pre> Required fields: ▪ id Body contains only fields that should be modified.
Success Response	Status code: 200 Content-type: application/json Body: <pre>[{ "id": <Number>, "name": <String>, "<custom field>": <Field type>, ... }, { "id": <Number>, "name": <String>, "<custom field>": <Field type>, ... } ...]</pre>
Error Response	400 (Invalid request) 401 (Unauthorized) 403 (Permission denied) 404 (Not Found) 422 (Validation Error) 500 (Internal Server Error)

Example	Request: <pre>curl -X POST http://localhost:9500/api/v1/positions/update -H "Content-Type: application/json" -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilsm" -d '[{ "id": 4,"name": "Head of HR"},{ "id": 5,"name": "System Administrator"}]'</pre> Response: <pre>[{ "id": 4, "name": "Head of HR" }, { "id": 5, "name": "System Administrator" }]</pre>
---------	---

2.3.8. Delete multiple positions

URL Endpoint	api/v1/positions/delete
Method	POST
Header Parameters	Authorization: Bearer <JWT access token> Content-Type: application/json
Path Parameters	
Query Parameters	
Body	[<Number>, <Number>] Body contains ids of positions that should be deleted.
Success Response	Status code: 200 Content-type: application/json Body: [<Number>, <Number>] Body contains ids of deleted positions.
Error Response	<u>400 (Invalid request)</u> <u>401 (Unauthorized)</u> <u>403 (Permission denied)</u> <u>404 (Not Found)</u> <u>422 (Validation Error)</u> <u>500 (Internal Server Error)</u>
Example	Request: curl -X POST http://localhost:9500/api/v1/positions/delete -H "Content-Type: application/json" -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilsm" -d '[4, 5]' Response: [4, 5]

2.4. Cards

2.4.1. Overview

The public REST API for managing cards provides the ability to have a card dictionary with the following formats:

- **UID** — allows to specify UID as a card value in raw format. It allows to assign this card in different hardware (Wiegand) formats using a specific algorithm. The algorithm is as follows: take the required number of bytes from the beginning (if there are not enough bytes provided, then fill with zeros before provided bytes) and reverse it.
Example:

- value — AABBCCDDEE
format — **W26** => value to assign — CCBBA
format — **W34** => value to assign — DDCCBBAA
- value — AABB
format — **W26** => value to assign — BBAA00
format — **W34** => value to assign — BBAA0000

- **MEMORY** — allows to provide a memory block that can be used for assigning this card in different formats. The algorithm is as follows: take the required number of bytes from the beginning.
Example:

- value — AABBCCDDEE
format — **W26** => value to assign — AABBC
format — **W34** => value to assign — AABBCDD

- Specific Wiegand format (**W26, W34, W36, W37, W42, W58DEC, W58**) — considers the value to be ready to assign.

The total number of identifiers assigned to an access object cannot exceed 5. For employees, this number includes both cards and virtual identifiers (QR codes, PINs, and mobile IDs).

"GuestApplicable" flag deprecation and backward compatibility

In previous software versions, cards with "guestApplicable"=true could be assigned to employees as "guest" cards.

Starting from software version 1.6.7.x, use dedicated endpoints for creating and assigning guest cards to employees and guests. This ensures full functionality, including guest card history on the "Guests" tab of the Client tool.

The "guestApplicable" flag remains for backward compatibility. Creating cards with "guestApplicable"=true or changing the flag from false to true is no longer supported. Cards with "guestApplicable"=true can still be used, but it is recommended to manage guest cards via the new endpoints.

If such cards exist, you can filter and:

- Change their flag to false, or
- Delete and re-add them via the "Guest Cards" endpoints.

Changing the flag or deleting a card is not allowed if the card is assigned.

When creating a new guest card with POST /api/v1/guestcards, the system checks for duplicates. If an unassigned card with "guestApplicable"=true and matching "value" and "format" exists, it will be automatically linked to the new guest card (i.e., cardholder of the "Guest badge" type).

2.4.2. Get all cards

URL Endpoint	api/v1/cards
Method	GET
Header Parameters	Authorization: Bearer <JWT access token>
Path Parameters	
Query Parameters	id – identifier(s) of Card(s) – returns only records with specified identifiers (e.g. api/v1/cards?id=1,2 or api/v1/cards?id=1&id=2) issued – if true, returns assigned cards, otherwise returns unassigned (free) cards (e.g. api/v1/cards?issued=true) name – filter by card's name with condition "starts with" (e.g. api/v1/cards?name=Card1) value – filter by card's value with condition "starts with" (e.g. api/v1/cards?value=AABBCC) search – filter by card's name or value or wiegand codekey (e.g. api/v1/cards?search=18AABBCC – search by codekey) guestApplicable – this flag is deprecated and preserved only for backward compatibility. Follow the link to learn more (e.g. api/v1/cards?guestApplicable=true) <any custom field> – e.g. custom field "holderName" has been created (e.g. api/v1/cards?holderName=135) excludeLegacyGuests – If true, cards issued to legacy guests are not taken into account (e.g. api/v1/cards?excludeLegacyGuests=true) limit – specifies how many entities the response should contain (e.g. api/v1/cards?limit=10 – means that only 10 entities are returned). The value of this parameter must be between 0 and 3000 offset – specifies how many entities should be skipped from the top (e.g. api/v1/cards?limit=10&offset=10 – means that 10 records are skipped, and next 10 entities are returned). The value of this parameter must be greater than or equal to 0 includeFields – specifies what fields the response should contain (e.g. api/v1/cards?includeFields=value – only the "value" field is returned) excludeFields – specifies what fields the response should not contain (e.g. api/v1/cards?excludeFields=id – all fields except "id" are returned) sortBy – specifies sort field (e.g. api/v1/cards?sortBy=name – means that entities would be sorted by the "name" field) sortOrder – specifies sort order. Can be either ASC or DESC (e.g. api/v1/cards?sortBy=name&sortOrder=DESC – means that entities would be sorted by the "name" field in descending order)
Body	
Success Response	Status code: 200 Content-type: application/json Body: <pre>[{ "id": <Number>, "name": <String>, "value": <String>,// card number in HEX "formattedValue": <String>,// formatted card number "format": <Enum: [UID, MEMORY, W26, W34, W36, W37, W42, W58, W58DEC]>, "holder": <{Holder object}>,// cardholder information "guestApplicable": <Boolean>,// deprecated; kept for backward compatibility "mfUid": <String>,// full Mifare UID "<custom field>": <Field type> }, ...]</pre> Field "value" contains HEX string of card's identifier where every 2 characters represent a hexadecimal number of 1 byte of a card identifier.

Error Response	<u>400 (Invalid request)</u> <u>401 (Unauthorized)</u> <u>403 (Permission denied)</u> <u>422 (Validation Error)</u> <u>500 (Internal Server Error)</u>
Example	Request: <pre>curl -X GET http://192.168.0.108:9500/api/v1/cards -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilsm"</pre> Response: <pre>[{ "id": 1, "name": "Card 1", "value": "600733", "formattedValue": "096,01843", "format": "W26", "holder": { "holderId": 10, "type": "EMP" }, "guestApplicable": false }]</pre>

2.4.3. Get card by ID

URL Endpoint	api/v1/cards/{id}
Method	GET
Header Parameters	Authorization: Bearer <JWT access token>
Path Parameters	id - unique ID of Card
Query Parameters	includeFields - specifies what fields the response should contain (e.g. api/v1/cards/1?includeFields=value - only field value is returned) excludeFields - specifies what fields the response should not contain (e.g. api/v1/cards/1?excludeFields=id - all fields except id are returned)
Body	
Success Response	Status code: 200 Content-type: application/json Body: <pre>{ "id": <Number>, "name": <String> "value": <String>, // card number in HEX "formattedValue": <String>, // formatted card number "format": <Enum: [UID, MEMORY, W26, W34, W36, W37, W42, W58, W58DEC]>, "holder": <{Holder object}>, // cardholder information "mfUid": <String>, // full Mifare UID "guestApplicable": <Boolean>, // deprecated; kept for backward compatibility "<custom field>": <Field type> }</pre>
Error Response	400 (Invalid request) 401 (Unauthorized) 403 (Permission denied) 404 (Card with specified id not found) 500 (Internal Server Error)
Example	Request: <pre>curl -X GET http://192.168.0.108:9500/api/v1/cards/1 -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilslm"</pre> Response: <pre>{ "id":1, "name":"Card 1", "value":"600733", "formattedValue": "096,01843", "format": "W26", "holder":{ "holderId":10, "type":"EMP" }, "guestApplicable": false }</pre>

2.4.4. Create card

URL Endpoint	api/v1/cards
Method	POST
Header Parameters	Authorization: Bearer <JWT access token> Content-Type: application/json
Path Parameters	
Query Parameters	
Body	<pre>{ "name": <String>, "value": <String>, "format": <Enum: [UID, MEMORY, W26, W34, W36, W37, W42, W58, W58DEC]>, "mfUid": <String>, "<custom field>": <Field type> }</pre> Required fields: - value — contains HEX string of card's identifier where every 2 characters represent a hexadecimal number of 1 byte of a card identifier - format Note: Specifying the "mfUid" field should be done with caution. Incorrectly entered "mfUid" values may cause the card to stop working or create duplicate UIDs. Responsibility for such issues lies with the user.
Success Response	Status code: 200 Content-type: application/json Body: <pre>{ "id": <Number>, "name": <String>, "value": <String>, "formattedValue": <String>, "format": <Enum: [UID, MEMORY, W26, W34, W36, W37, W42, W58, W58DEC]>, "mfUid": <String>, "guestApplicable": <Boolean>, "<custom field>": <Field type> }</pre>
Error Response	<u>400 (Invalid Request)</u> <u>401 (Unauthorized)</u> <u>403 (Permission denied)</u> <u>422 (Validation Error)</u> <u>500 (Internal Server Error)</u>
Example	Request: <pre>curl -X POST http://192.168.0.108:9500/api/v1/cards -H "Content-Type: application/json" -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilsm" -d '{"name": "Card 1", "value": "ABCCDDEE", "format": "W26"}'</pre> Response: <pre>{ "id": 2, "name": "Card 1", "value": "ABCCDDEE", "formattedValue": "171,52445", "format": "W26", "guestApplicable": false }</pre>

2.4.5. Update card

URL Endpoint	api/v1/cards/{id}
Method	PUT
Header Parameters	Authorization: Bearer <JWT access token>
Path Parameters	id — unique ID of a Card
Query Parameters	
Body	<pre>{ "name": <String>, "<custom field>": <Field type>, "mfUid": <String> }</pre> Body contains only fields that should be modified. It is possible to modify the following fields: - name <custom_field> - mfUid — a HEX value Note: Updating the "mfUid" field should be done with caution. Incorrectly entered "mfUid" values may cause the card to stop working or create duplicate UIDs. Responsibility for such issues lies with the user.
Success Response	Status code: 200 Content-type: application/json Body: <pre>{ "id": <Number>, "name": <String>, "value": <String>, "formattedValue": <String>, "format": <Enum: [UID, MEMORY, W26, W34, W36, W37, W42, W58, W58DEC]>, "holder": <{Holder object}>, "mfUid": <String>, "guestApplicable": <Boolean>, "<custom field>": <Field type>, }</pre>
Error Response	<u>400 (Invalid request)</u> <u>401 (Unauthorized)</u> <u>403 (Permission denied)</u> <u>404 (Card with specified ID not found)</u> <u>422 (Validation Error)</u> <u>500 (Internal Server Error)</u>
Example	Request: <pre>curl -X PUT http://192.168.0.108:9500/api/v1/cards/2 -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilslm" -d {"name":"Card 2"}</pre> Response: <pre>{ "id":2, "name":"Card 2", "value ":"600733", "formattedValue": "096,01843", "format": "W26", "holder": { "holderId": 10, "type": "EMP" }, "mfUid": "3307609f", "guestApplicable": false }</pre>

2.4.6. Delete card

URL Endpoint	api/v1/cards/{id}
Method	DELETE
Header Parameters	Authorization: Bearer <JWT access token>
Path Parameters	id - unique ID of Card
Query Parameters	
Body	
Success Response	Status code: 200 Content-type: application/json Body: <pre>{ "id": <Number>, "name": <String>, "value": <String>, "format": <Enum: [UID, MEMORY, W26, W34, W36, W37, W42, W58, W58DEC]>, "formattedValue": <String>, "mfUid": <String>, "guestApplicable": <Boolean>, "<custom field>": <Field type> }</pre>
Error Response	<u>400 (Invalid request)</u> <u>401 (Unauthorized)</u> <u>403 (Permission denied)</u> <u>404 (Card with specified id not found)</u> <u>422 (Validation Error)</u> <u>500 (Internal Server Error)</u>
Example	Request: <pre>curl -X DELETE http://192.168.0.108:9500/api/v1/cards/2 -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilslm"</pre> Response: <pre>{ "id": 2, "name": "Card 1", "value": "ABCCDDEE", "formattedValue": "171,52445", "format": "W26", "guestApplicable": false }</pre>

2.4.7. Create multiple cards

URL Endpoint	api/v1/cards/create
Method	POST
Header Parameters	Authorization: Bearer <JWT access token> Content-Type: application/json
Path Parameters	
Query Parameters	
Body	<pre>[{ "name": <String>, "value": <String>, "format": <Enum: [UID, MEMORY, W26, W34, W36, W37, W42, W58, W58DEC]>, "mfUid": <String> "guestApplicable": <Boolean>, "<custom field>": <Field type> ... }, { "name": <String>, "value": <String>, "format": <Enum: [UID, MEMORY, W26, W34, W36, W37, W42, W58, W58DEC]>, "mfUid": <String> "guestApplicable": <Boolean>, "<custom field>": <Field type> ... } ...]</pre> Required fields: ▪ valueformat Field "value" contains HEX string of card's identifier where every 2 characters represent a hexadecimal number of 1 byte of a card identifier. Note: Specifying the "mfUid" field should be done with caution. Incorrectly entered "mfUid" values may cause the card to stop working or create duplicate UIDs. Responsibility for such issues lies with the user.

Success Response	Status code: 200 Content-type: application/json Body: <pre>[{ "id": <Number>, "name": <String>, "value": <String>, "formattedValue": <String>, "format": <Enum: [UID, MEMORY, W26, W34, W36, W37, W42, W58, W58DEC]>, "mfUid": <String>, "guestApplicable": <Boolean>, "<custom field>": <Field type> ... }, { "id": <Number>, "name": <String>, "value": <String>, "formattedValue": <String>, "format": <Enum: [UID, MEMORY, W26, W34, W36, W37, W42, W58, W58DEC]>, "mfUid": <String>, "guestApplicable": <Boolean>, "<custom field>": <Field type> ... } ...]</pre>
Error Response	400 (Invalid Request) 401 (Unauthorized) 403 (Permission denied) 422 (Validation Error) 500 (Internal Server Error)
Example	Request: <pre>curl -X POST http://localhost:9500/api/v1/cards/create -H "Content-Type: application/json" -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilsm" -d '{"name": "Card 1", "value": "ABCCDDEE", "format": "W26"}, {"name": "Card 2", "value": "BBCCDDFF", "format": "W26"}'</pre> Response: <pre>[{ "id": 1, "name": "Card 1", "value": "ABCCDDEE", "formattedValue": "171,52445", "format": "W26", "guestApplicable": false }, { "id": 2, "name": "Card 2", "value": "BBCCDDFF", "formattedValue": "187,52445", "format": "W26", "guestApplicable": false }]</pre>

2.4.8. Update multiple cards

URL Endpoint	api/v1/cards/update
Method	POST
Header Parameters	Authorization: Bearer <JWT access token> Content-Type: application/json
Path Parameters	
Query Parameters	
Body	<pre>[{ "id": <Number>, "name": <String>, "mfUid": <String>, "<custom field>": <Field type> ... }, { "id": <Number>, "name": <String>, "mfUid": <String>, "<custom field>": <Field type> ... } ...]</pre> Required fields: ▪ id Body contains fields that should be updated. It is possible to modify the following fields: ▪ name ▪ <custom_field> ▪ mfUid — a HEX value If you send "name":"" in the body of the request, the card name will be deleted (it will become an empty string). If you send "name": null, the card name will not be changed. Note: Updating the "mfUid" field should be done with caution. Incorrectly entered "mfUid" values may cause the card to stop working or create duplicate UIDs. Responsibility for such issues lies with the user.

Success Response	Status code: 200 Content-type: application/json Body: <pre>[{ "id": <Number>, "name": <String>, "value": <String>, "formattedValue": <String>, "format": <Enum: [UID, MEMORY, W26, W34, W36, W37, W42, W58, W58DEC]>, "mfUid": <String>, "guestApplicable": <Boolean>, "<custom field>": <Field type> }, { "id": <Number>, "name": <String>, "value": <String>, "formattedValue": <String>, "format": <Enum: [UID, MEMORY, W26, W34, W36, W37, W42, W58, W58DEC]>, "mfUid": <String>, "guestApplicable": <Boolean>, "<custom field>": <Field type> }, ...]</pre>
Error Response	400 (Invalid Request) 401 (Unauthorized) 403 (Permission denied) 404 (Not Found) 422 (Validation Error) 500 (Internal Server Error)
Example	Request: <pre>curl -X POST http://localhost:9500/api/v1/cards/update -H "Content-Type: application/json" -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlciIsbm" -d '{"id":1,"name":"Card 1 updated"},"id":2,"name":"Card 2 updated"}]</pre> Response: <pre>[{ "id": 1, "name": "Card 1 updated", "value": "ABCCDDEE", "formattedValue": "171,52445", "format": "W26", "guestApplicable": false }, { "id": 2, "name": "Card 2 updated", "value": "BBCCDDFF", "formattedValue": "187,52445", "format": "W26", "guestApplicable": false }]</pre>

2.4.9. Delete multiple cards

URL Endpoint	api/v1/cards/delete
Method	POST
Header Parameters	Authorization: Bearer <JWT access token> Content-Type: application/json
Path Parameters	
Query Parameters	
Body	[<Number>, <Number>] Body contains IDs of cards that should be deleted.
Success Response	Status code: 200 Content-type: application/json Body: [<Number>, <Number>] Body contains IDs of deleted cards.
Error Response	<u>400 (Invalid Request)</u> <u>401 (Unauthorized)</u> <u>403 (Permission denied)</u> <u>404 (Not Found)</u> <u>422 (Validation Error)</u> <u>500 (Internal Server Error)</u>
Example	Request: curl -X POST http://localhost:9500/api/v1/cards/delete -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilsm" -d '[1, 2]' Response: [1, 2]

2.5. Employees

2.5.1. Get all employees

Returns a list of employees subject to the user's permissions.

URL Endpoint	api/v1/employees
Method	GET
Header Parameters	Authorization: Bearer <JWT access token>
Path Parameters	
Query Parameters	▪ id – employee identifier(s). Returns only records with the specified identifiers (e.g. api/v1/employees?id=1,2 or api/v1/employees?id=1&id=2) ▪ departmentId – filter by department identifiers. To return employees that are not assigned to any department, use departmentId=0. You can specify multiple IDs (e.g. api/v1/employees?departmentId=1,2) ▪ positionId – filter by position identifiers. You can specify multiple IDs (e.g. api/v1/employees?positionId=2, api/v1/employees?positionId=0,1,2) ▪ name – filter by employee name with a "starts with" comparison (e.g. api/v1/employees?name=Jack) ▪ blocked – specifies whether to return only blocked (true) or only unblocked (false) employees (e.g. api/v1/employees?blocked=true) ▪ tabId[operation] – filter by personnel number. Possible values for [operation]: EQ, STARTS_WITH. If no operation is specified, EQ is used by default (e.g. api/v1/employees?tabId%5BSTARTS_WITH%5D=4533454) ▪ accessStartTime[operation] – access period start time (ISO 8601). Possible values for [operation]: EQ, GT, LT, GTE, LTE (e.g. api/v1/employees?accessStartTime%5BEQ%5D=2025-07-29T10:15:30) ▪ accessEndTime[operation] – access period end time (ISO 8601). Possible values for [operation]: EQ, GT, LT, GTE, LTE (e.g. api/v1/employees?accessEndTime%5BEQ%5D=2025-07-29T10:15:30) ▪ includeQrImage – specifies whether to return a Base64-encoded image of the QR code (false by default) (e.g. api/v1/employees?includeQrImage=true) ▪ limit – specifies how many entities the response should contain (e.g. api/v1/employees?limit=10 – means that only 10 entities are returned). The value of this parameter must be greater than or equal to 0. There is no entity limit for this request. ▪ offset – specifies how many entities should be skipped from the top (e.g. api/v1/employees?limit=10&offset=10 – means that 10 records are skipped, and next 10 entities are returned). The value of this parameter must be greater than or equal to 0 ▪ sortBy – specifies the sort field (e.g. api/v1/employees?sortBy=name – means that entities are sorted by the "name" field) ▪ sortOrder – specifies the sort order. Can be either ASC or DESC (e.g. api/v1/employees?sortBy=name&sortOrder=DESC – means that entities would be sorted by "name" field in descending order) ▪ <custom field>[operation] – filter by custom fields. Possible values for [operation]: IN, EQ, NEQ, GT, LT, GTE, LTE, IS_NULL, IS_NOT_NULL, STARTS_WITH, CONTAINS (e.g. api/v1/employees?externalId%5BCONTAINS%5D=12) ▪ includeFields – specifies what fields the response should contain (e.g. api/v1/employees?includeFields=name,tabId – only the "name" and "tabId" fields are returned) ▪ excludeFields – specifies what fields should be omitted from the response (e.g. api/v1/employees?excludeFields=contactDetails – all fields except "contactDetails" are returned) ▪ isGlobal – specifies whether to return only global (true) or only local (false) employees. All employees are returned if not specified (default) (e.g. api/v1/employees?isGlobal=true)
Body	

Success Response	Status code: 200 Content-type: application/json Body: <pre>[{ "id": <Number>, // employee identifier "name": <String>, // employee name "departmentId": <Number>, // department identifier "departmentName": <String>, // department name "positionId": <Number>, // position identifier "positionName": <String>, // position name "vehicles": [// employee vehicles assigned to the employee { "id": <Number>, // vehicle identifier "lpNumber": <String>, // license plate number "verificationPin": <String>, // vehicle verification PIN (used in scenarios such as "Card+PIN", etc.) "<custom field>": <Field type>, // vehicle custom field ... }, ...], "photo": <String>, // employee photo (Base64-encoded) "<custom field>": <Field type>, // employee custom field ..., "contactDetails": [// contacts used for notifications { "type": <Enum: [SMS, VIBER, TELEGRAM, EMAIL, MAX]>, // notification channel "value": <String>, // phone number, username, or email address "preferable": <Boolean> // preferred notification type }, ...], "isBlocked": <Boolean>, // whether the employee is blocked from access "description": <String>, // free-form text "tabId": <String>, // personnel number "location": { // where the employee was last registered "zoneId": <Number>, // zone identifier "zoneName": <String>, // zone name "entranceTime": <DateTime> // when the employee entered the zone }, "verificationPin": <String>, // verification PIN (used in authentication scenarios such as "Card+PIN", etc.) "accessStartTime": <DateTime>, // access period start "accessEndTime": <DateTime>, // access period end "identifiers": [// virtual identifiers { "type": <Enum: [QR, PIN, MOBILE_ID]>, // identifier type ... }], "isGlobal": <Boolean> // whether the employee is global or local }, ...]</pre>
Error Response	400 (Invalid Request) 401 (Unauthorized) 403 (Access Denied) 422 (Validation Error) 500 (Internal Server Error)

Example	Request: <pre>curl -X 'GET' \ 'http://192.168.0.108:9500/api/v1/employees?includeQrImage=true&sortBy=name' \ -H 'Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlciIsIm'</pre> Response: <pre>[{ "id": 43, "name": "Jack Dawson", "departmentId": 47, "departmentName": "Sales Department", "positionId": 1, "positionName": "Sales Manager", "photo": "/9j/4AAQSkZJRgABAgAAQAABAAD/2wBDAA", "contactDetails": [{ "type": "SMS", "value": "+1234567890", "preferable": false }, { "type": "VIBER", "value": "+1234567890", "preferable": false }, { "type": "TELEGRAM", "value": "@telegramUsername", "preferable": false }, { "type": "EMAIL", "value": "example@example.com", "preferable": true }, { "type": "MAX", "value": "123456789", "preferable": false }], "isBlocked": false, "tabId": "0123-4", "location": { "zoneId": 4, "zoneName": "Hallway", "entranceTime": "2025-10-10 13:28:00" }, "verificationPin": "9843", "accessStartTime": "2025-10-10T11:37:00", "accessEndTime": "2025-10-31T11:37:00", "vehicles": [{ "id": 48, "lpNumber": "AB123CD", "verificationPin": "1409" }], "identifiers": [{ "type": "PIN", "value": "002478", "formattedValue": "9336" }], "isGlobal": false }]</pre>
---------	--

2.5.2. Get employee by ID

Returns the employee with the specified ID, subject to the user's permissions.

URL Endpoint	api/v1/employees/{id}
Method	GET
Header Parameters	Authorization: Bearer <JWT access token>
Path Parameters	▪ id – employee identifier
Query Parameters	▪ withFullPhoto – specifies whether the response contains the employee's full photo (false by default) (e.g. api/v1/employees/1?withFullPhoto=true) ▪ includeFields – specifies what fields the response should contain (e.g. api/v1/employees/1?includeFields=name,departmentName – only the "name" and "departmentName" fields are returned) ▪ excludeFields – specifies what fields should be omitted from the response (e.g. api/v1/employees/1?excludeFields=contactDetails – all fields except "contactDetails" are returned)
Body	

Success Response	Status code: 200 Content-type: application/json Body: <pre>[{ "id": <Number>, // employee identifier "name": <String>, // employee name "departmentId": <Number>, // department identifier "departmentName": <String>, // department name "positionId": <Number>, // position identifier "positionName": <String>, // position name "vehicles": [// employee vehicles assigned to employee { "id": <Number>, // vehicle identifier "lpNumber": <String>, // license plate number "verificationPin": <String>, // vehicle verification PIN (used in authentication scenarios such as "Card+PIN", etc.) "<custom field>": <Field type>, // vehicle custom field }, ...], "photo": <String>, // employee photo (Base64-encoded) "<custom field>": <Field type>, // employee custom field }, ...], "contactDetails": [// contacts used for notifications { "type": <Enum: [SMS, VIBER, TELEGRAM, EMAIL, MAX]>, // notification channel "value": <String>, // phone number, username, or email address "preferable": <Boolean> // preferred notification type }, ...], "isBlocked": <Boolean>, // whether the employee is blocked from access "description": <String>, // free-form text "tabId": <String>, // personnel number "location": { // where the employee was last registered "zoneId": <Number>, // zone identifier "zoneName": <String>, // zone name "entranceTime": <DateTime> // when the employee entered the zone }, "verificationPin": <String>, // verification PIN (used in scenarios such as "Card+PIN", etc.) "accessStartTime": <DateTime>, // access period start "accessEndTime": <DateTime>, // access period end "identifiers": [// virtual identifiers { "type": <Enum: [QR, PIN, MOBILE_ID]>, ... }, ...], "isGlobal": <Boolean> // whether the employee is global or local }]</pre>
Error Response	<u>400 (Invalid Request)</u> <u>401 (Unauthorized)</u> <u>403 (Access Denied)</u> <u>404 (Not Found)</u> <u>500 (Internal Server Error)</u>
	Request: <pre>curl -X 'GET' \ 'http://192.168.0.108:9500/api/v1/employees/43' \ -H 'Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlciIsIm'</pre> Response: <pre>{ "id": 43, "name": "Jack Dawson", "departmentId": 47, "departmentName": "Sales Department", "positionId": 1, "positionName": "Sales Manager",</pre>

Example	<pre>"vehicles": [{ "id": 48, "lpNumber": "AB123CD", "verificationPin": "1409" }], "photo": "/9j/4AAQSkZJRgABAgAAAQABAAD/2wBDAA", "contactDetails": [{ "type": "SMS", "value": "+1234567890", "preferable": false }, { "type": "VIBER", "value": "+1234567890", "preferable": false }, { "type": "TELEGRAM", "value": "@telegramUsername", "preferable": false }, { "type": "EMAIL", "value": "example@example.com", "preferable": true }, { "type": "MAX", "value": "123456789", "preferable": false }], "isBlocked": false, "tblId": "0123-4", "location": { "zoneld": 4, "zoneName": "Hallway", "entranceTime": "2025-10-10 13:28:00" }, "verificationPin": "9843", "accessStartTime": "2025-10-10T11:37:00", "accessEndTime": "2025-10-31T11:37:00", "identifiers": [{ "type": "PIN", "value": "002478", "formattedValue": "9336" }, { "type": "QR", "format": "W26", "value": "3F2BB3", "formattedValue": "063,11187", "encodedValue": "3F2BB3", "image": "iVBORw0KGgoAAAANSUHEUgAAAPoAAAD", "startTime": "2025-10-14T08:00:00", "expirationTime": "2025-10-18T23:59:00" }], "isGlobal": false}</pre>
---------	--

2.5.3. Count employees

URL Endpoint	api/v1/employees/count
Method	GET
Header Parameters	Authorization: Bearer <JWT access token>
Path Parameters	
Query Parameters	id – employee identifier(s). Returns the number of records with the specified identifiers (e.g. api/v1/employees/count?id=1,2 or api/v1/employees/count?id=1&id=2) departmentId – filter by department identifiers. To count employees that are not assigned to any department, use departmentId=0. You can specify multiple IDs (e.g. api/v1/employees/count?departmentId=1) positionId – filter by position identifiers. You can specify multiple IDs (e.g. api/v1/employees/count?positionId=1) name – filter by employee name with a "starts with" comparison (e.g. api/v1/employees/count?name=IvanIvanov) blocked – specifies whether to count only blocked (true) or only unblocked (false) employees (e.g. api/v1/employees/count?blocked=true) <custom field> – filter by a custom field using an "equals" comparison (e.g. api/v1/employees/count?externalId=135) isGlobal – specifies whether to count only global (true) or only local (false) employees. All employees are counted if not specified (default) (e.g. api/v1/employees/count?isGlobal=true) groupBy – specifies which field to use for grouping. With groupBy set, the result is a list of per-group counts instead of a single total. Possible values: departmentId, departmentName, positionId, positionName, name (e.g. api/v1/employees/count?groupBy=departmentName) isRecursive – whether to count employees recursively (e.g. api/v1/employees/count?isRecursive=true)
Body	
Success Response	Status code: 200 Content-type: application/json Body: <pre>[{ "groupByField": <String>, "employeeCount": <Number> }, ...]</pre>
Error Response	401 (Unauthorized) 403 (Access Denied) 422 (Validation Error) 500 (Internal Server Error)
Example 1	Request: <pre>curl -X 'GET' \ 'http://192.168.0.108:9500/api/v1/employees/count' \ -H 'Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlciIsIm'</pre> Response: <pre>[{ "groupByField": null, "employeeCount": 14 }]</pre>

Example 2	Request: <pre>curl -X 'GET' \ 'http://192.168.0.108:9500/api/v1/employees/count?groupBy=departmentId' \ -H 'Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1Y1lljoiVXNlcilslm'</pre> Response: <pre>[{ "groupByField": 0, "employeeCount": 1 }, { "groupByField": 14, "employeeCount": 4 }, { "groupByField": 15, "employeeCount": 3 }, { "groupByField": 16, "employeeCount": 6 }]</pre>
-----------	--

2.5.4. Create employee

URL Endpoint	api/v1/employees
Method	POST
Header Parameters	Authorization: Bearer <JWT access token> Content-Type: application/json
Path Parameters	
Query Parameters	bindAccessPoints – when true (the default), the employee is bound to the access points available to the calling user. Pass false to create the employee without access point bindings (e.g. /api/v1/employees?bindAccessPoints=false)
Body	<pre>{ "name": <String>, "departmentId": <Number>, "departmentName": <String>, "positionId": <Number>, "positionName": <String>, "photo": <String>, "custom field": <Field type>, ..., "contactDetails": [{ "type": <Enum: [SMS, VIBER, TELEGRAM, EMAIL, MAX]>, "value": <String>, "preferable": <Boolean> }, ...], "description": <String>, "tabId": <String>, "verificationPin": <String>, "accessStartTime": <DateTime>, "accessEndTime": <DateTime> }</pre> Required fields: - name - departmentId Optional fields: - verificationPin – a sequence of decimal digits (4 digits, 0000–9999). Automatically set if not specified - accessStartTime – Employee access period start (ISO 8601). If not set, the current date and time are used - accessEndTime – Employee access period end (ISO 8601). If not set, the access period is unlimited

Success Response	Status code: 200 Content-type: application/json Body: <pre>{ "id": <Number>, "name": <String>, "departmentId": <Number>, "departmentName": <String>, "positionId": <Number>, "positionName": <String>, "photo": <String>, "custom field": <Field type>, ..., "contactDetails": [{ "type": <Enum: [SMS, VIBER, TELEGRAM, EMAIL, MAX]>, "value": <String>, "preferable": <Boolean> }, ...], "isBlocked": <Boolean>, "description": <String>, "tabId": <String>, "verificationPin": <String>, "accessStartTime": <DateTime>, "accessEndTime": <DateTime>, "isGlobal": <Boolean> }</pre>
Error Response	<u>401 (Unauthorized)</u> <u>403 (Permission Denied)</u> <u>422 (Validation Error)</u> <u>500 (Internal Server Error)</u>

Example	Request: <pre>curl -X 'POST' \ 'http://192.168.0.108:9500/api/v1/employees' \ -H 'Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilslm' \ -H 'Content-Type: application/json' \ -d '{ "name": "Tom Taylor", "departmentId": 47, "positionId": 1 }'</pre> Response: <pre>{ "id": 51, "name": "Tom Taylor", "departmentId": 47, "departmentName": "Sales Department", "positionId": 1, "positionName": "Sales Manager", "contactDetails": [{ "type": "SMS", "value": null, "preferable": false }, { "type": "VIBER", "value": null, "preferable": false }, { "type": "TELEGRAM", "value": null, "preferable": false }, { "type": "EMAIL", "value": null, "preferable": false }, { "type": "MAX", "value": null, "preferable": false }], "isBlocked": false, "verificationPin": "8043", "accessStartTime": "2025-10-14T12:07:19.1528845", "isGlobal": false }</pre>
---------	---

2.5.5. Update employee

URL Endpoint	api/v1/employees/{id}
Method	PUT
Header Parameters	Authorization: Bearer <JWT access token> Content-Type: application/json
Path Parameters	id – employee identifier
Query Parameters	
Body	<pre>{ "name": <String>, "departmentId": <Number>, "positionId": <Number>, "photo": <String>, "<custom field>": <Field type>, ..., "contactDetails": [{ "type": <Enum: [SMS, VIBER, TELEGRAM, EMAIL, MAX]>, "value": <String>, "preferable": <Boolean> }], "description": <String>, "tabId": <String>, "verificationPin": <String>, "accessStartTime": <DateTime>, "accessEndTime": <DateTime> }</pre> Body contains fields that should be modified.

Success Response	Status code: 200 Content-type: application/json Body: <pre>{ "id": <Number>, "name": <String>, "departmentId": <Number>, "departmentName": <String>, "positionId": <Number>, "positionName": <String>, "photo": <String>, "<custom field>": <Field type>, ..., "vehicles": [{ "id": <Number>, "lpNumber": <String>, "<custom field>": <Field type>, ... }, ...], "contactDetails": [{ "type": <Enum: [SMS, VIBER, TELEGRAM, EMAIL, MAX]>, "value": <String>, "preferable": <Boolean> }, ...], "isBlocked": <Boolean>, "description": <String>, "tabId": <String>, "location": { "zoneId": <Number>, "zoneName": <String>, "entranceTime": <DateTime> }, "verificationPin": <String>, "accessStartTime": <DateTime>, "accessEndTime": <DateTime>, "identifiers": [{ "type": <Enum: [QR, PIN, MOBILE_ID]>, ... }], "isGlobal": <Boolean> }</pre>
Error Response	401 (Unauthorized) 403 (Access Denied) 404 (Not Found) 422 (Validation Error) 500 (Internal Server Error)

Example	Request: <pre>curl -X 'PUT' \ 'http://192.168.0.108:9500/api/v1/employees/2' \ -H 'Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1Y1lljoiVXNlcilslm' \ -H 'Content-Type: application/json' \ -d '{ "name": "Tom Taylor" }'</pre> Response: <pre>{ "id": 2, "name": "Tom Taylor", "departmentId": 47, "departmentName": "Sales Department", "positionId": 1, "positionName": "Sales Manager", "photo": "/9j/4AAQSkZJRgABAgAAAQABAAD/2wBDAA", "contactDetails": [{ "type": "SMS", "value": "+1234567890", "preferable": false }, { "type": "VIBER", "value": "+1234567890", "preferable": false }, { "type": "TELEGRAM", "value": "@telegramUsername", "preferable": false }, { "type": "EMAIL", "value": "example@example.com", "preferable": true }, { "type": "MAX", "value": "123456789", "preferable": false }], "isBlocked": false, "tabId": "0123-4", "location": { "zoneId": 4, "zoneName": "Hallway", "entranceTime": "2025-10-10 13:28:00" }, "verificationPin": "9843", "accessStartTime": "2025-10-10T11:37:00", "accessEndTime": "2025-10-31T11:37:00", "isGlobal": false }</pre>
---------	--

2.5.6. Delete employee

URL Endpoint	api/v1/employees/{id}
Method	DELETE
Header Parameters	Authorization: Bearer <JWT access token>
Path Parameters	id – employee identifier
Query Parameters	
Body	
Success Response	Status code: 200 Content-type: application/json Body: <pre>[{ "id": <Number>, "name": <String>, "departmentId": <Number>, "departmentName": <String>, "positionId": <Number>, "positionName": <String>, "vehicles": [{ "id": <Number>, "lpNumber": <String>, "verificationPin": <String>, "<custom field>": <Field type>, ... }, ...], "photo": <String>, "<custom field>": <Field type>, ..., "contactDetails": [{ "type": <Enum: [SMS, VIBER, TELEGRAM, EMAIL, MAX]>, "value": <String>, "preferable": <Boolean> }, ...], "isBlocked": <Boolean>, "description": <String>, "tabId": <String>, "location": { "zoneId": <Number>, "zoneName": <String>, "entranceTime": <DateTime> }, "verificationPin": <String>, "accessStartTime": <DateTime>, "accessEndTime": <DateTime>, "identifiers": [{ "type": <Enum: [QR, PIN, MOBILE_ID]>, ... }], "isGlobal": <Boolean> }]</pre>

Error Response	401 (Unauthorized) 403 (Access denied) 404 (Not Found) 422 (Validation Error) 500 (Internal Server Error)
Example	Request: curl -X 'DELETE' \ 'http://192.168.0.108:9500/api/v1/employees/2' \ -H 'Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilslm' Response: [{ "id": 2, "name": "Tom Taylor", "departmentId": 3, "departmentName": "Hardware R&D", "positionId": 9, "positionName": "QA", "photo": "/9j/4AAQSkZJRgABA...", "contactDetails": [{ "type": "SMS", "value": "", "preferable": true }, { "type": "VIBER", "value": "", "preferable": false }, { "type": "TELEGRAM", "value": "", "preferable": false }, { "type": "EMAIL", "value": "", "preferable": false }, { "type": "MAX", "value": "", "preferable": false }], "isBlocked": false, "description": "", "tabId": "12345", "location": { "zoneId": 0, "zoneName": "outer territory", "entranceTime": "2022-07-27 17:29:49" }, "verificationPin": "1392", "accessStartTime": "2022-07-25T13:07:45", "isGlobal": false }]

2.5.7. Block employee

URL Endpoint	api/v1/employees/{id}/block
Method	PUT
Header Parameters	Authorization: Bearer <JWT access token>
Path Parameters	id – employee identifier
Query Parameters	
Body	
Success Response	Status code: 204 No content
Error Response	<u>401 (Unauthorized)</u> <u>403 (Access Denied)</u> <u>404 (Not Found)</u> <u>500 (Internal Server Error)</u>
Example	Request: <pre>curl -X 'PUT' \ 'http://192.168.0.108:9500/api/v1/employees/6/block' \ -H 'Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlciIsIm'</pre> Response: 204 No content

2.5.8. Unblock employee

URL Endpoint	api/v1/employees/{id}/unblock
Method	PUT
Header Parameters	Authorization: Bearer <JWT access token>
Path Parameters	id – employee identifier
Query Parameters	
Body	
Success Response	Status code: 204 No content
Error Response	401 (Unauthorized) 403 (Access Denied) 404 (Not Found) 500 (Internal Server Error)
Example	Request: <pre>curl -X 'PUT' \ 'http://192.168.0.108:9500/api/v1/employees/6/unblock' \ -H 'Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlciIsIm'</pre> Response: 204 No content

2.5.9. Create multiple employees

URL Endpoint	api/v1/employees/create
Method	POST
Header Parameters	Authorization: Bearer <JWT access token> Content-Type: application/json
Path Parameters	
Query Parameters	bindAccessPoints – when true (the default), the employees are bound to the access points available to the calling user. Pass false to create the employees without access point bindings (e.g. /api/v1/employees/create?bindAccessPoints=false)
Body	<pre>[{ "name": <String>, "departmentId": <Number>, "departmentName": <String>, "positionId": <Number>, "positionName": <String>, "photo": <String>, "<custom field>": <Field type>, ..., "contactDetails": [{ "type": <Enum: [SMS, VIBER, TELEGRAM, EMAIL, MAX]>, "value": <String>, "preferable": <Boolean> }, ...], "description": <String>, "tabId": <String>, "verificationPin": <String>, "accessStartTime": <DateTime>, "accessEndTime": <DateTime> }, ...]</pre> Required fields: - name - departmentId

Success Response	Status code: 200 Content-type: application/json Body: <pre>[{ "id": <Number>, "name": <String>, "departmentId": <Number>, "departmentName": <String>, "positionId": <Number>, "positionName": <String>, "photo": <String>, "<custom field>": <Field type>, ..., "contactDetails": [{ "type": <Enum: [SMS, VIBER, TELEGRAM, EMAIL, MAX]>, "value": <String>, "preferable": <Boolean> }, ...], "isBlocked": <Boolean>, "description": <String>, "tabId":<String>, "verificationPin": <String>, "accessStartTime": <DateTime>, "accessEndTime": <DateTime>, "isGlobal": <Boolean> }, ...]</pre>
Error Response	<u>401 (Unauthorized)</u> <u>403 (Access Denied)</u> <u>422 (Validation Error)</u> <u>500 (Internal Server Error)</u>

Example	Request: <pre>curl -X 'POST' \ 'http://192.168.0.108:9500/api/v1/employees/create' \ -H 'Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilsm' \ -H 'Content-Type: application/json' \ -d '[{"name":"Tom Taylor","departmentId":3,"positionId":9}, {"name":"Jack Dawson","departmentId":3,"positionId":10}]'</pre> Response: <pre>[{ "id": 99, "name": "Tom Taylor", "departmentId": 3, "departmentName": "Hardware R&D", "positionId": 9, "positionName": "QA", "contactDetails": [{"type": "SMS", "value": null, "preferable": false}, {"type": "VIBER", "value": null, "preferable": false}, {"type": "TELEGRAM", "value": null, "preferable": false}, {"type": "EMAIL", "value": null, "preferable": false}, {"type": "MAX", "value": null, "preferable": false}], "isBlocked": false, "verificationPin": "8043", "accessStartTime": "2025-10-14T12:07:19.1528845", "isGlobal": false }, { "id": 100, "name": "Jack Dawson", "departmentId": 3, "departmentName": "Hardware R&D", "positionId": 10, "positionName": "HW Developer", "contactDetails": [{"type": "SMS", "value": null, "preferable": false}, {"type": "VIBER", "value": null, "preferable": false}, {"type": "TELEGRAM", "value": null, "preferable": false}, {"type": "EMAIL", "value": null, "preferable": false}, {"type": "MAX", "value": null, "preferable": false}], "isBlocked": false, "verificationPin": "2374", "accessStartTime": "2025-10-14T12:07:19.1528845", "isGlobal": false }]</pre>
---------	---

2.5.10. Update multiple employees

URL Endpoint	api/v1/employees/update
Method	POST
Header Parameters	Authorization: Bearer <JWT access token> Content-Type: application/json
Path Parameters	
Query Parameters	
Body	<pre>[{ "id": <Number>, "name": <String>, "departmentId": <Number>, "positionId": <Number>, "photo": <String>, "<custom field>": <Field type>, ..., "contactDetails": [{ "type": <Enum: [SMS, VIBER, TELEGRAM, EMAIL, MAX]>, "value": <String>, "preferable": <Boolean> }], "description": <String>, "tabId": <String>, "verificationPin": <String>, "accessStartTime": <DateTime>, "accessEndTime": <DateTime> } ...]</pre> Required fields: id Body contains only fields that should be modified.

Success Response	Status code: 200 Content-type: application/json Body: <pre>[{ "id": <Number>, "name": <String>, "departmentId": <Number>, "departmentName": <String>, "positionId": <Number>, "positionName": <String>, "photo": <String>, "<custom field>": <Field type>, ... "vehicles": [{ "id": <Number>, "IpNumber": <String>, "verificationPin": <String>, "<custom field>": <Field type>, ... }, ...], "contactDetails": [{ "type": <Enum: [SMS, VIBER, TELEGRAM, EMAIL, MAX]>, "value": <String>, "preferable": <Boolean> }, ...], "isBlocked": <Boolean>, "description": <String>, "tabId":<String>, "location": { "zoneId": <Number>, "zoneName": <String>, "entranceTime": <DateTime> }, "verificationPin": <String>, "accessStartTime": <DateTime>, "accessEndTime": <DateTime>, "identifiers": [{ "type": <Enum: [QR, PIN, MOBILE_ID]>, ... }], "isGlobal": <Boolean> }, ...]</pre>
Error Response	401 (Unauthorized) 403 (Access Denied) 404 (Not Found) 422 (Validation Error) 500 (Internal Server Error)

Example	Request: <pre>curl -X 'POST' \ 'http://192.168.0.108:9500/api/v1/employees/update' \ -H 'Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilslm' \ -H 'Content-Type: application/json' \ -d '{\"id\":99,\"name\":\"Tomas Taylor\"},{\"id\":100,\"name\":\"David Bowie\"}]'</pre> Response: <pre>[{ \"id\": 99, \"name\": \"Tomas Taylor\", \"departmentId\": 3, \"departmentName\": \"Hardware R&D\", \"positionId\": 9, \"positionName\": \"QA\", \"contactDetails\": [{\"type\": \"SMS\", \"value\": null, \"preferable\": false}, {\"type\": \"VIBER\", \"value\": null, \"preferable\": false}, {\"type\": \"TELEGRAM\", \"value\": null, \"preferable\": false}, {\"type\": \"EMAIL\", \"value\": null, \"preferable\": false}, {\"type\": \"MAX\", \"value\": null, \"preferable\": false}], \"isBlocked\": false, \"verificationPin\": \"8043\", \"accessStartTime\": \"2025-10-14T05:07:19\", \"isGlobal\": false }, { \"id\": 100, \"name\": \"David Bowie\", \"departmentId\": 3, \"departmentName\": \"Hardware R&D\", \"positionId\": 10, \"positionName\": \"HW Developer\", \"contactDetails\": [{\"type\": \"SMS\", \"value\": null, \"preferable\": false}, {\"type\": \"VIBER\", \"value\": null, \"preferable\": false}, {\"type\": \"TELEGRAM\", \"value\": null, \"preferable\": false}, {\"type\": \"EMAIL\", \"value\": null, \"preferable\": false}, {\"type\": \"MAX\", \"value\": null, \"preferable\": false}], \"isBlocked\": false, \"verificationPin\": \"1234\", \"accessStartTime\": \"2025-10-14T05:07:19\", \"isGlobal\": false }]</pre>
---------	--

2.5.11. Delete multiple Employees

URL Endpoint	api/v1/employees/delete
Method	POST
Header Parameters	Authorization: Bearer <JWT access token>
Path Parameters	
Query Parameters	
Body	[<Number>, <Number>, ...] Body contains IDs of employees to be deleted.
Success Response	Status code: 200 Content-type: application/json Body: [<Number>, <Number>, ...]
Error Response	401 (Unauthorized) 403 (Access Denied) 404 (Not Found) 422 (Validation Error) 500 (Internal Server Error)
Example	Request: <pre>curl -X 'POST' \ 'http://192.168.0.108:9500/api/v1/employees/delete' \ -H 'Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlciIsIm' \ -d '[99,100]'</pre> Response: <pre>[99,100]</pre>

2.5.12. Block multiple employees

URL Endpoint	api/v1/employees/block
Method	POST
Header Parameters	Authorization: Bearer <JWT access token>
Path Parameters	
Query Parameters	
Body	[<Number>, <Number> ...] Body contains IDs of employees to be blocked.
Success Response	Status code: 204 No content
Error Response	<u>401 (Unauthorized)</u> <u>403 (Access Denied)</u> <u>404 (Not Found)</u> <u>500 (Internal Server Error)</u>
Example	Request: <pre>curl -X 'POST' \ 'http://192.168.0.108:9500/api/v1/employees/block' \ -H 'Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1Y1lljoiVXNlciIsIm' \ -d '[6, 9]'</pre> Response: 204 No content

2.5.13. Unblock multiple employees

URL Endpoint	api/v1/employees/unblock
Method	POST
Header Parameters	Authorization: Bearer <JWT access token> Content-Type: application/json
Path Parameters	
Query Parameters	
Body	[<Number>, <Number> ...] Body contains IDs of employees to be unblocked.
Success Response	Status code: 204 No content
Error Response	<u>401 (Unauthorized)</u> <u>403 (Access Denied)</u> <u>404 (Not Found)</u> <u>500 (Internal Server Error)</u>
Example	Request: <pre>curl -X 'POST' \ 'http://192.168.0.108:9500/api/v1/employees/unblock' \ -H 'Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilslm' \ -d '[6, 9]'</pre> Response: 204 No content

2.6. Official vehicles

2.6.1. Get all official vehicles

URL Endpoint	api/v1/vehicles/official
Method	GET
Header Parameters	Authorization: Bearer <JWT access token>
Path Parameters	
Query Parameters	id – filters by Official Vehicle ID(s). Returns only records matching the specified IDs. (e.g. api/v1/vehicles/official?id=1,2 or api/v1/vehicles/official?id=1&id=2) departmentId – filters by Department IDs. To get Official Vehicles that are not assigned to any department, use departmentId=0. Multiple IDs can be specified. (e.g. api/v1/vehicles/official?departmentId=2, api/v1/vehicles/official?departmentId=0,1,2) lpNumber – filters by the Official Vehicle's license plate number using a "starts with" condition. (e.g. api/v1/vehicles/official?lpNumber=s123) limit – specifies how many entities the response should contain. The value of this parameter must be between 0 and 3000. (e.g. api/v1/vehicles/official?limit=10 – only 10 entities are returned) offset – specifies how many entities should be skipped from the top. The value of this parameter must be greater than or equal to 0. (e.g. api/v1/vehicles/official?limit=10&offset=10 – the first 10 records are skipped, and the next 10 entities are returned) <any custom field> – filters by the value of a custom field. (e.g. custom field "externalId": api/v1/vehicles/official?externalId=135) includeFields – returns only the specified fields. (e.g. api/v1/vehicles/official?includeFields=lpNumber – only the "lpNumber" field is returned) excludeFields – excludes the specified fields from the response. (e.g. api/v1/vehicles/official?excludeFields=id – all fields except "id" are returned) sortBy – specifies the field to sort by. (e.g. api/v1/vehicles/official?sortBy=lpNumber – entities are sorted by the "lpNumber" field) sortOrder – specifies the sort order. Can be either ASC or DESC. (e.g. api/v1/vehicles/official?sortBy=lpNumber&sortOrder=DESC – entities are sorted by the "lpNumber" field in descending order)
Body	
Success Response	Status code: 200 Content-type: application/json Body: <pre>[{ "id": <Number>, "lpNumber": <String>, "departmentId": <Number>, "departmentName": <String>, "<custom field>": <Field type>, "verificationPin": <String>,// verification PIN (used in authentication scenarios such as "Card+PIN", etc.) "accessStartTime": <DateTime>,// vehicle access period start "accessEndTime": <DateTime> // vehicle access period end }, ...]</pre>
Error Response	<u>400 (Invalid request)</u> <u>401 (Unauthorized)</u> <u>403 (Permission denied)</u> <u>422 (Validation Error)</u> <u>500 (Internal Server Error)</u>

Example	Request: <pre>curl -X GET http://192.168.0.108:9500/api/v1/vehicles/official -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilsm"</pre> Response: <pre>[{ "id": 17, "lpNumber": "a123aa", "departmentId": 4, "departmentName": "Official vehicles", "verificationPin": "4995", "accessStartTime": "2022-06-15T00:00:00" }</pre>
---------	---

2.6.2. Get official vehicle by ID

URL Endpoint	api/v1/vehicles/official/{id}
Method	GET
Header Parameters	Authorization: Bearer <JWT access token>
Path Parameters	id - identifier of official vehicle
Query Parameters	includeFields - specifies what fields the response should contain (e.g. api/v1/vehicles/official/1?includeFields=lpNumber - only field lpNumber is returned in the response) excludeFields - specifies what fields the response should not contain (e.g. api/v1/official/vehicles/1?excludeFields=id - all fields except id are returned in the response)
Body	
Success Response	Status code: 200 Content-type: application/json Body: <pre>{ "id": <Number>, "lpNumber": <String>, "departmentId": <Number>, "departmentName": <String>, "<custom field>": <Field type>, "verificationPin": <String>, // verification PIN (used in authentication scenarios such as "Card+PIN", etc.) "accessStartTime": <DateTime>, // vehicle access period start "accessEndTime": <DateTime> // vehicle access period end }</pre>
Error Response	<u>400 (Invalid request)</u> <u>401 (Unauthorized)</u> <u>403 (Permission denied)</u> <u>404 (Department with specified id not found)</u> <u>500 (Internal Server Error)</u>
Example	Request: <pre>curl -X GET http://192.168.0.108:9500/api/v1/vehicles/official/17 -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilsm"</pre> Response: <pre>{ "id": 17, "lpNumber": "a123aa", "departmentId": 4, "departmentName": "Official vehicles", "verificationPin": "4995", "accessStartTime": "2022-06-15T00:00:00" }</pre>

2.6.3. Create official vehicle

URL Endpoint	api/v1/vehicles/official
Method	POST
Header Parameters	Authorization: Bearer <JWT access token> Content-Type: application/json
Path Parameters	
Query Parameters	
Body	<pre>{ "lpNumber": <String>, "departmentId": <Number>, "<custom field>": <Field type>, "verificationPin": <String>, "accessStartTime": <DateTime>, "accessEndTime": <DateTime> }</pre> Required fields: - lpNumber Optional fields: - verificationPin — a sequence of decimal digits (4 digits, 0000–9999). Automatically set if not specified - accessStartTime — Official Vehicle access period start (ISO 8601). If not set, the current date and time are used - accessEndTime — Official Vehicle access period end (ISO 8601). If not set, the access period is unlimited
Success Response	Status code: 200 Content-type: application/json Body: <pre>{ "id": <Number>, "lpNumber": <String>, "departmentId": <Number>, "departmentName": <String>, "<custom field>": <Field type>, "verificationPin": <String>, "accessStartTime": <DateTime>, "accessEndTime": <DateTime> }</pre>
Error Response	400 (Invalid Request) 401 (Unauthorized) 403 (Permission Denied) 404 (Department With Specified ID Not Found) 422 (Validation Error) 500 (Internal Server Error)
Example	Request: <pre>curl -X POST http://192.168.0.108:9500/api/v1/vehicles/official -H "Content-Type: application/json" -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjojVXNlcilsm" -d '{"lpNumber":"r586po"}'</pre> Response: <pre>{ "id":3, "lpNumber ":"r586po", "departmentId": 0, "verificationPin": "5174", "accessStartTime": "2025-10-13T15:16:26" }</pre>

2.6.4. Update official vehicle

URL Endpoint	api/v1/vehicles/official/{id}
Method	PUT
Header Parameters	Authorization: Bearer <JWT access token> Content-Type: application/json
Path Parameters	id – identifier of Official vehicle to be edited
Query Parameters	
Body	<pre>{ "lpNumber": <String>, "departmentId": <Number>, "<custom field>": <Field type>, "verificationPin": <String>, "accessStartTime": <DateTime>, "accessEndTime": <DateTime> }</pre> Body contains only fields that should be modified.
Success Response	Status code: 200 Content-type: application/json Body: <pre>{ "id": <Number>, "lpNumber": <String>, "departmentId": <Number>, "departmentName": <String>, "<custom field>": <Field type>, "verificationPin": <String>, "accessStartTime": <DateTime>, "accessEndTime": <DateTime> }</pre>
Error Response	400 (Invalid Request) 401 (Unauthorized) 403 (Permission Denied) 404 (Department With The Specified ID Not Found) 422 (Validation Error) 500 (Internal Server Error)
Example	Request: <pre>curl -X PUT http://192.168.0.108:9500/api/v1/vehicles/official/3 -H "Content-Type: application/json" -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlciIsbmIiOiJpNumber": "AB123CD"}</pre> Response: <pre>{ "id": 3, "lpNumber": "AB123CD", "departmentId": 4, "departmentName": "Official vehicles", "verificationPin": "5174", "accessStartTime": "2025-10-13T15:16:26" }</pre>

2.6.5. Delete official vehicle

URL Endpoint	api/v1/vehicles/official/{id}
Method	DELETE
Header Parameters	Authorization: Bearer <JWT access token>
Path Parameters	id – identifier of Official vehicle to be deleted
Query Parameters	
Body	
Success Response	Status code: 200 Content-type: application/json Body: <pre>{ "id": <Number>, "lpNumber": <String>, "departmentId": <Number>, "departmentName": <String>, "<custom field>": <Field type>, "verificationPin": <String>, "accessStartTime": <DateTime>, "accessEndTime": <DateTime> }</pre>
Error Response	400 (Invalid request) 401 (Unauthorized) 403 (Permission denied) 404 (Department with specified id not found) 422 (Validation Error) 500 (Internal Server Error)
Example	Request: <pre>curl -X DELETE http://192.168.0.108:9500/api/v1/vehicles/official/17 -H "Content-Type: application/json" -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilslm"</pre> Response: <pre>{ "id": 17, "lpNumber": "a123aa", "departmentId": 4, "departmentName": "Official vehicles", "verificationPin": "4995", "accessStartTime": "2022-06-15T00:00:00" }</pre>

2.6.6. Create multiple official vehicles

URL Endpoint	api/v1/vehicles/official/create
Method	POST
Header Parameters	Authorization: Bearer <JWT access token> Content-Type: application/json
Path Parameters	
Query Parameters	
Body	<pre>[{ "lpNumber": <String>, "departmentId": <Number>, "<custom field>": <Field type>, "verificationPin": <String>, "accessStartTime": <DateTime>, "accessEndTime": <DateTime> }, { "lpNumber": <String>, "departmentId": <Number>, "<custom field>": <Field type>, "verificationPin": <String>, "accessStartTime": <DateTime>, "accessEndTime": <DateTime> } ...]</pre> Required fields: lpNumber
Success Response	Status code: 200 Content-type: application/json Body: <pre>[{ "id": <Number>, "lpNumber": <String>, "departmentId": <Number>, "departmentName": <String>, "<custom field>": <Field type>, "verificationPin": <String>, "accessStartTime": <DateTime>, "accessEndTime": <DateTime> }, { "id": <Number>, "lpNumber": <String>, "departmentId": <Number>, "departmentName": <String>, "<custom field>": <Field type>, "verificationPin": <String>, "accessStartTime": <DateTime>, "accessEndTime": <DateTime> } ...]</pre>
Error Response	400 (Invalid Request) 401 (Unauthorized) 403 (Permission Denied) 404 (Not Found) 422 (Validation Error) 500 (Internal Server Error)

Example	Request: <pre>curl -X POST http://localhost:9500/api/v1/vehicles/official/ create -H "Content-Type: application/json" -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilsm" -d '{"lpNumber":"a111aa"},"{"lpNumber":"b222bb"}'</pre> Response: <pre>[{ "id": 4, "lpNumber": "a111aa", "departmentId": 0, "verificationPin": "5174", "accessStartTime": "2025-10-13T15:16:26" }, { "id": 5, "lpNumber": "b222bb", "departmentId": 0, "verificationPin": "7654", "accessStartTime": "2025-10-13T15:16:26" }]</pre>
---------	---

2.6.7. Update multiple official vehicles

URL Endpoint	api/v1/vehicles/official/update
Method	POST
Header Parameters	Authorization: Bearer <JWT access token> Content-Type: application/json
Path Parameters	
Query Parameters	
Body	<pre>[{ "IpNumber": <String>, "departmentId": <Number>, "<custom field>": <Field type>, "verificationPin": <String>, "accessStartTime": <DateTime>, "accessEndTime": <DateTime> }, { "IpNumber": <String>, "departmentId": <Number>, "<custom field>": <Field type>, "verificationPin": <String>, "accessStartTime": <DateTime>, "accessEndTime": <DateTime> }, ...]</pre> Required fields: ▪ id Body contains only fields that should be modified.
Success Response	Status code: 200 Content-type: application/json Body: <pre>[{ "id": <Number>, "IpNumber": <String>, "departmentId": <Number>, "departmentName": <String>, "<custom field>": <Field type>, "verificationPin": <String>, "accessStartTime": <DateTime>, "accessEndTime": <DateTime> }, { "id": <Number>, "IpNumber": <String>, "departmentId": <Number>, "departmentName": <String>, "<custom field>": <Field type>, "verificationPin": <String>, "accessStartTime": <DateTime>, "accessEndTime": <DateTime> } ...]</pre>
Error Response	<u>400 (Invalid Request)</u> <u>401 (Unauthorized)</u> <u>403 (Permission Denied)</u> <u>404 (Not Found)</u> <u>422 (Validation Error)</u> <u>500 (Internal Server Error)</u>

Example	Request: <pre>curl -X POST http://localhost:9500/api/v1/vehicles/official/ update -H "Content-Type: application/json" -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilslm" -d '[{ "id": 4, "lpNumber": "a888aa"}, { "id": 5, "lpNumber": "b999bb"}]'</pre> Response: <pre>[{ "id": 4, "lpNumber": "a888aa", "departmentId": 0, "verificationPin": "5174", "accessStartTime": "2025-10-10T08:00:00" }, { "id": 5, "lpNumber": "b999bb", "departmentId": 0, "verificationPin": "7364", "accessStartTime": "2025-10-12T15:00:00" }]</pre>
---------	--

2.6.8. Delete multiple official vehicles

URL Endpoint	api/v1/vehicles/official/delete
Method	POST
Header Parameters	Authorization: Bearer <JWT access token> Content-Type: application/json
Path Parameters	
Query Parameters	
Body	[<Number>, <Number>] Body contains IDs of official vehicles that should be deleted.
Success Response	Status code: 200 Content-type: application/json Body: [<Number>, <Number>] Body contains IDs of deleted official vehicles.
Error Response	<u>400 (Invalid request)</u> <u>401 (Unauthorized)</u> <u>403 (Permission denied)</u> <u>404 (Not Found)</u> <u>500 (Internal Server Error)</u>
Example	Request: curl -X POST http://localhost:9500/api/v1/vehicles/official/delete -H "Content-Type: application/json" -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilsm" -d '[4, 5]' Response: [4, 5]

2.7. Deprecated: Employee vehicles

2.7.1. Add new employee vehicle

This endpoint is deprecated. Use employee-related [custom fields](#) to store employee vehicle registration numbers.

URL Endpoint	api/v1/employees/{id}/vehicles
Method	POST
Header Parameters	Authorization: Bearer <JWT access token> Content-Type: application/json
Path Parameters	id - identifier of an employee
Query Parameters	
Body	<pre>{ "lpNumber": <String>, "departmentId": <Number>, "<custom field>": <Field type>, ... }</pre> Required fields: lpNumber
Success Response	Status code: 200 Content-type: application/json Body: <pre>{ "id": <Number>, "lpNumber": <String>, "<custom field>": <Field type>, ... }</pre>
Error Response	401 (Unauthorized) 403 (Permission denied) 404 (Employee with specified id not found) 422 (Validation Error) 500 (Internal Server Error)
Example	Request: <pre>curl -X POST http://192.168.0.108:9500/api/v1/employees/1/vehicles -H "Content-Type: application/json" -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilsm" -d '{"lpNumber": "r586po"}'</pre> Response: <pre>{ "id": 3, "lpNumber": "r586po" }</pre>

2.7.2. Update employee vehicle

This endpoint is deprecated. Use employee-related [custom fields](#) to store employee vehicle registration numbers.

URL Endpoint	api/v1/employees/{id}/vehicles/{vehicleId}
Method	PUT
Header Parameters	Authorization: Bearer <JWT access token> Content-Type: application/json
Path Parameters	id - identifier of an employee vehicleId - identifier of Vehicle to be edited
Query Parameters	
Body	<pre>{ "id": <Number>, "lpNumber": <String>, "departmentId": <Number>, "<custom field>": <Field type>, ... }</pre> Body contains only modified fields.
Success Response	Status code: 200 Content-type: application/json Body: <pre>{ "id": <Number>, "lpNumber": <String>, "<custom field>": <Field type>, ... }</pre>
Error Response	401 (Unauthorized) 403 (Permission denied) 404 (Employee with specified id not found) 422 (Validation Error) 500 (Internal Server Error)
Example	Request: <pre>curl -X PUT http://192.168.0.108:9500/api/v1/employees/1/vehicles/3 -H "Content-Type: application/json" -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilsm" -d '{"lpNumber": "r586po"}'</pre> Response: <pre>{ "id": 3, "lpNumber": "r586po" }</pre>

2.7.3. Delete employee vehicle

This endpoint is deprecated. Use employee-related [custom fields](#) to store employee vehicle registration numbers.

URL Endpoint	api/v1/employees/{id}/vehicles/{vehicleId}
Method	DELETE
Header Parameters	Authorization: Bearer <JWT access token>
Path Parameters	id - identifier of an employee vehicleId - identifier of an employee vehicle to be deleted
Query Parameters	
Body	
Success Response	Status code: 200 Content-type: application/json Body: <pre>{ "id": <Number>, "lpNumber": <String>, "<custom field>": <Field type>, ... }</pre>
Error Response	401 (Unauthorized) 403 (Permission denied) 404 (Employee with specified id not found) 422 (Validation Error) 500 (Internal Server Error)
Example	Request: <pre>curl -X DELETE http://192.168.0.108:9500/api/v1/employees/1/vehicles/3 -H "Content-Type: application/json" -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilslm"</pre> Response: <pre>{ "id": 3, "lpNumber": "r586po" }</pre>

2.8. Bindings

2.8.1. Employees-Cards

Get Employee - Card bindings

URL Endpoint	api/v1/bindings/employees-cards
Method	GET
Header Parameters	Authorization: Bearer <JWT access token>
Path Parameters	
Query Parameters	employeeId - identifier(s) of Employee(s) - returns only records with specified identifiers (e.g. api/v1/bindings/employees-cards?employeeId=1,2 or api/v1/bindings/employees-cards?employeeId=1&employeeId=2) cardId - IDs of Card(s) - returns only records with specified identifiers (e.g. api/v1/bindings/employees-cards?cardId=1,2 or api/v1/bindings/employees-cards?cardId=1&cardId=2) limit - specifies how many entities the response should contain (e.g. api/v1/bindings/employees-cards?limit=10 - means that only 10 entities returned in a response). The value of this parameter must be between 0 and 3000 offset - specifies how many entities should be skipped from the top (e.g. api/v1/bindings/employees-cards?limit=10&offset=10 - means that 10 records are skipped and next 10 entities returned in a response). The value of this parameter must be greater than or equal to 0
Body	
Success Response	Status code: 200 Content-type: application/json Body: <pre>[{ "employeeId": <Number>, "cardId": <Number>, "startDate": <Date>, // card validity start "expirationDate": <Date>, // card validity end "format": <Enum: [W26, W34, W36, W37, W42, W58, W58DEC]> }, ...]</pre>
Error Response	<u>400 (Invalid request)</u> <u>401 (Unauthorized)</u> <u>403 (Permission denied)</u> <u>500 (Internal Server Error)</u>
Example	Request: <pre>curl -X GET http://192.168.0.108:9500/api/v1/bindings/employees-cards -H "Content-Type: application/json" -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilsm"</pre> Response: <pre>[{ "employeeId": 1, "cardId": 1, "startDate": "2023-12-03T17:57:09.104Z", "expirationDate": "2023-12-03T17:57:09.104Z", "format": "W26" }]</pre>

Assign Cards to Employees

URL Endpoint	api/v1/bindings/employees-cards
Method	POST
Header Parameters	Authorization: Bearer <JWT access token> Content-Type: application/json
Path Parameters	
Query Parameters	
Body	<pre>[{ "employeeId": <Number>, "cardId": <Number>, "startDate": <Date>, "expirationDate": <Date>, "format": <Enum: [W26, W34, W36, W37, W42, W58, W58DEC]> }, ...]</pre> Required fields: • employeeId • cardId • format
Success Response	Status code: 204 No Content
Error Response	<u>400 (Invalid request)</u> <u>401 (Unauthorized)</u> <u>403 (Permission denied)</u> <u>422 (Validation Error)</u> <u>500 (Internal Server Error)</u>
Example	Request: <pre>curl -X POST http://192.168.0.108:9500/api/v1/bindings/employees-cards -H "Content-Type: application/json" -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilsm"-d '[{ "employeeId": 1, "cardId": 2, "format": "W26", "startDate": "2021-03-01 18:00:00", "expirationDate": "2021-03-31 18:00:00"}]'</pre> Response: 204 No Content

Update expiration time for the specified Cards assigned to Employees

URL Endpoint	api/v1/bindings/employees-cards
Method	PUT
Header Parameters	Authorization: Bearer <JWT access token> Content-Type: application/json
Path Parameters	
Query Parameters	
Body	<pre>[{ "employeeId": <Number>, "cardId": <Number>, "expirationDate": <Date>, "format": <Enum: [W26, W34, W36, W37, W42, W58, W58DEC]> }, ...]</pre> Required fields: • employeeId • cardId • expirationDate If "expirationDate" was specified earlier, and now you do not send its value (or send {"expirationDate": null}), then the values of both "startDate" and "expirationDate" become null.
Success Response	Status code: 204 No Content
Error Response	400 (Invalid request) 401 (Unauthorized) 403 (Permission denied) 422 (Validation Error) 500 (Internal Server Error)
Example	Request: <pre>curl -X PUT http://192.168.0.108:9500/api/v1/bindings/employees-cards -H "Content-Type: application/json" -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilsm" -d '{"employeeId": 1, "cardId": 2, "format": "W26", "expirationDate": "2021-03-31 18:00:00"}'</pre> Response: 204 No Content

Update start and expiration time for the specified Cards assigned to Employees

URL Endpoint	api/v1/bindings/employees-cards
Method	PATCH
Header Parameters	Authorization: Bearer <JWT access token> Content-Type: application/json
Path Parameters	
Query Parameters	
Body	<pre>[{ "employeeId":<Number>, "cardId": <Number>, "startDate":<Date>, "expirationDate": <Date>, "format": <Enum: [W26, W34, W36, W37, W42, W58, W58DEC]> }, ...]</pre> Required fields: - employeeId - cardId - startDate - expirationDate If you send an empty value of "startDate" or/and "expirationDate", then the value of a parameter becomes null.
Success Response	Status code: 204 No Content
Error Response	400 (Invalid request) 401 (Unauthorized) 403 (Permission denied) 422 (Validation Error) 500 (Internal Server Error)
Example	Request: <pre>curl -X PATCH http://192.168.0.108:9500/api/v1/bindings/employees-cards -H "Content-Type: application/json" -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilsm" -d '{"employeeId":1, "cardId": 2, "format": "W26", "startDate":"2021-03-01 18:00:00", "expirationDate":"2021-03-31 18:00:00"}'</pre> Response: 204 No Content

Remove Cards from Employees

URL Endpoint	api/v1/bindings/employees-cards/delete
Method	POST
Header Parameters	Authorization: Bearer <JWT access token> Content-Type: application/json
Path Parameters	
Query Parameters	
Body	<pre>[{ "employeeId": <Number>, "cardId": <Number>, "startDate": <Date>, "expirationDate": <Date>, "format": <Enum: [W26, W34, W36, W37, W42, W58, W58DEC]> }, ...]</pre> Required fields: - employeeId - cardId - format
Success Response	Status code: 204 No Content
Error Response	400 (Invalid request) 401 (Unauthorized) 403 (Permission denied) 422 (Validation Error) 500 (Internal Server Error)
Example	Request: <pre>curl -X POST http://192.168.0.108:9500/api/v1/bindings/employees-cards/delete -H "Content-Type: application/json" -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilsm" -d '[{ "employeeId": 1, "cardId": 2, "startDate": "2023-12-03T19:03:22.155Z", "expirationDate": "2023-12-03T19:03:22.155Z", "format": "W26"}]'</pre> Response: 204 No Content

2.8.2. Employees-Guest Cards

Get Employee-Guest Card bindings

URL Endpoint	/api/v1/bindings/employees-guestcards
Method	GET
Header Parameters	Authorization: Bearer <JWT access token>
Path Parameters	
Query Parameters	employeeId — identifier(s) of Employee(s). Returns only records with the specified identifiers (e.g. api/v1/bindings/employees-guestcards?employeeId=1,2 or api/v1/bindings/employees-guestcards?employeeId=1&employeeId=2) guestCardId — identifier(s) of Guest Card(s). Returns only records with the specified identifiers (e.g. api/v1/bindings/employees-guestcards?guestCardId=1,2 or api/v1/bindings/employees-guestcards?guestCardId=1&guestCardId=2) startTime[operation] — filter by start date and time. Possible values for [operation]: EQ, GT, LT, GTE, LTE. (e.g. api/v1/bindings/employees-guestcards?startTime%5Bgte%5D=2023-05-10T00:00:00Z) expirationTime[operation] — filter by end date and time. Possible values for [operation]: EQ, GT, LT, GTE, LTE. (e.g. api/v1/bindings/employees-guestcards?expirationTime%5Bgte%5D=2023-05-10T00:00:00Z) - Both ISO 8601 and YYYY-MM-DD hh:mm:ss date-time formats are accepted in requests, but the response is always returned in ISO 8601 format. includeFields — specifies what fields the response should contain (e.g. api/v1/bindings/employees-guestcards?includeFields=guestCardId,description — only fields "guestCardId" and "description" are returned) excludeFields — specifies what fields should be omitted from the response (e.g. api/v1/bindings/employees-guestcards?excludeFields=description — all fields except "description" are returned) limit — specifies how many entities the response should contain (e.g. api/v1/bindings/employees-guestcards?limit=10 — means that only 10 entities are returned). The value of this parameter must be between 0 and 3000 offset — specifies how many entities should be skipped from the top (e.g. api/v1/bindings/employees-guestcards?limit=10&offset=10 — means that 10 records are skipped, and next 10 entities are returned). The value of this parameter must be greater than or equal to 0 sortBy — specifies sort field (e.g. /api/v1/bindings/employees-guestcards?sortBy=startTime — means that entities would be sorted by "startTime" field) sortOrder — specifies sort order. Can be either ASC or DESC (e.g. api/v1/bindings/employees-guestcards?sortBy=expirationTime&sortOrder=desc — means that entities would be sorted by "expirationTime" field in descending order)
Body	
Success Response	Status code: 200 Content-type: application/json Body: <pre>[{ "employeeId": <Number>, "guestCardId": <Number>, "startTime": <DateTime>, // guest card validity start "expirationTime": <DateTime>, // guest card validity end "description": <String> }, ...]</pre>
Error Response	400 (Invalid request) 401 (Unauthorized) 403 (Permission denied) 500 (Internal Server Error)

Example	Request: <pre>curl -X GET http://192.168.0.108:9500/api/v1/bindings/employees-guestcards -H "Content-Type: application/json" -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilsm"</pre> Response: <pre>[{ "employeeId": 8, "guestCardId": 9, "startTime": "2025-08-22T11:01:00", "expirationTime": "2025-08-22T19:01:00", "description": "Guest badge for Building #1" }]</pre>
---------	--

Assign Guest Cards to Employees

URL Endpoint	api/v1/bindings/employees-guestcards
Method	POST
Header Parameters	Authorization: Bearer <JWT access token> Content-Type: application/json
Path Parameters	
Query Parameters	replace — defines how to handle a Guest Card if it is already assigned. Possible values: - true — reassigns the Guest Card to another Employee, closing the previous entry with the current expiration date - false (default) — returns an error message (HTTP 422) (e.g. api/v1/bindings/employees-guestcards?replace=true)
Body	<pre>[{ "employeeId": <Number>, "guestCardId": <Number>, "startTime": <DateTime>, "expirationTime": <DateTime>, "description": <String> }, ...]</pre> Required fields: - employeeId - guestCardId Optional fields: - description - startTime — if not set, the current date and time are used - expirationTime — if not set, it is calculated as startTime plus the "Default guest badge expiration time" value configured in the "Client" tool ("File — Settings — Guest badges") Both ISO 8601 and YYYY-MM-DD hh:mm:ss date-time formats are accepted in requests.
Success Response	Status code: 204 No Content
Error Response	400 (Invalid request) 401 (Unauthorized) 403 (Permission denied) 422 (Validation Error) 500 (Internal Server Error)
Example	Request: <pre>curl -X POST http://192.168.0.108:9500/api/v1/bindings/employees-guestcards/assign?replace={false} -H "Content-Type: application/json" -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlciIsIm" -d '{"employeeId": 1, "guestCardId": 2, "startTime": "2025-12-03T17:57:09.104Z", "expirationTime": "2025-12-10T17:57:09.104Z", "description": "Guest badge for Building #1"}'</pre> Response: 204 No Content

Update Guest Cards assigned to Employees

URL Endpoint	api/v1/bindings/employees-guestcards/update
Method	POST
Header Parameters	Authorization: Bearer <JWT access token> Content-Type: application/json
Path Parameters	
Query Parameters	
Body	<pre>[{ "employeeId": <Number>, "guestCardId": <Number>, "startTime": <DateTime>, "expirationTime": <DateTime>, "description": <String> }, ...]</pre> Required fields: - employeeId - guestCardId Body contains only fields that should be modified. Both ISO 8601 and YYYY-MM-DD hh:mm:ss date-time formats are accepted in requests.
Success Response	Status code: 204 No Content
Error Response	400 (Invalid request) 401 (Unauthorized) 403 (Permission denied) 422 (Validation Error) 500 (Internal Server Error)
Example	Request: <pre>curl -X POST http://192.168.0.108:9500/api/v1/bindings/employees-guestcards/update -H "Content-Type: application/json" -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlciIsIm" -d '{"employeeId":1, "guestCardId": 2, "expirationTime":"2025-10-07T13:51:46.398Z"}'</pre> Response: 204 No Content

Withdraw Guest Cards assigned to Employees

URL Endpoint	api/v1/bindings/employees-guestcards/delete
Method	POST
Header Parameters	Authorization: Bearer <JWT access token> Content-Type: application/json
Path Parameters	
Query Parameters	
Body	<pre>[{ "employeeId":<Number>, "guestCardId": <Number>, }, ...]</pre> Required fields: - employeeId - guestCardId
Success Response	Status code: 204 No Content
Error Response	400 (Invalid request) 401 (Unauthorized) 403 (Permission denied) 422 (Validation Error) 500 (Internal Server Error)
Example	Request: <pre>curl -X POST http://192.168.0.108:9500/api/v1/bindings/employees-guestcards/delete -H "Content-Type: application/json" -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilsm" -d '[{ "employeeId":1, "cardId": 2}]'</pre> Response: 204 No Content

2.8.3. Employees-Access Rules

Get Employees Access Rules

URL Endpoint	api/v1/bindings/employees-accessrules
Method	GET
Header Parameters	Authorization: Bearer <JWT access token>
Path Parameters	
Query Parameters	employeeId — identifier(s) of Employee(s). Returns only records with specified identifiers (e.g. api/v1/bindings/employees-accessrules?employeeId=1,2 or api/v1/bindings/employees-accessrules?employeeId=1&employeeId=2) accessruleId — identifier(s) of Access Rule(s). Returns only records with specified identifiers (e.g. api/v1/binding/employees-accessrules?accessruleId=1,2 or api/v1/bindings/employees-accessrules?accessruleId=1&accessruleId=2) limit — specifies how many entities the response should contain (e.g. api/v1/bindings/employees-accessrules?limit=10 — means that only 10 entities are returned in a response). The value of this parameter must be between 0 and 3000 offset — specifies how many entities should be skipped from the top (e.g. api/v1/bindings/employees-accessrules?limit=10&offset=10 — means that 10 records are skipped, and next 10 entities are returned in a response). The value of this parameter must be greater than or equal to 0
Body	
Success Response	Status code: 200 Content-type: application/json Body: <pre>[{ "employeeId": <Number>, "accessruleId": <Number> }, ...]</pre>
Error Response	400 (Invalid request) 401 (Unauthorized) 403 (Permission denied) 500 (Internal Server Error)
Example	Request: <pre>curl -X GET http://192.168.0.108:9500/api/v1/bindings/employees-accessrules -H "Content-Type: application/json" -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlciIsIm"</pre> Response: <pre>[{ "employeeId": 1, "accessruleId": 1 }, { "employeeId": 1, "accessruleId": 2 }, { "employeeId": 2, "accessruleId": 1 }]</pre>

Assign Access Rules to Employees

URL Endpoint	api/v1/bindings/employees-accessrules
Method	POST
Header Parameters	Authorization: Bearer <JWT access token> Content-Type: application/json
Path Parameters	
Query Parameters	
Body	<pre>[{ "employeeId": <Number>, "accessruleId": <Number> }, ...]</pre> Required fields: - employeeId - accessruleId
Success Response	Status code: 204 No Content
Error Response	<u>400 (Invalid request)</u> <u>401 (Unauthorized)</u> <u>403 (Permission denied)</u> <u>422 (Validation Error)</u> <u>500 (Internal Server Error)</u>
Example	Request: <pre>curl -X POST http://192.168.0.108:9500/api/v1/bindings/employees-accessrules -H "Content-Type: application/json" -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilsm" -d '[{ "employeeId":1, "accessruleId":1}, {"employeeId":2, "accessruleId":2}]'</pre> Response: 204 No Content

Remove Access Rules from Employees

URL Endpoint	api/v1/bindings/employees-accessrules/delete
Method	POST
Header Parameters	Authorization: Bearer <JWT access token> Content-Type: application/json
Path Parameters	
Query Parameters	
Body	<pre>[{ "employeeId": <Number>, "accessruleId": <Number> }, ...]</pre> Required fields: - employeeId - accessruleId
Success Response	Status code: 204 No Content
Error Response	<u>400 (Invalid request)</u> <u>401 (Unauthorized)</u> <u>403 (Permission denied)</u> <u>422 (Validation Error)</u> <u>500 (Internal Server Error)</u>
Example	Request: <pre>curl -X POST http://192.168.0.108:9500/api/v1/bindings/employees-accessrules/delete -H "Content-Type: application/json" -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilsm" -d '{"employeeId":1, "accessruleId":1}, {"employeeId":2, "accessruleId":2}'</pre> Response: 204 No Content

2.8.4. Employees-Access Points

Get bindings between Employees and Access Points

URL Endpoint	api/v1/bindings/employees-accesspoints
Method	GET
Header Parameters	Authorization: Bearer <JWT access token>
Path Parameters	
Query Parameters	▪ employeeId - identifier(s) of Employee(s) - returns only records with specified identifiers (e.g. api/v1/bindings/employees-accesspoints?employeeId=1,2 or api/v1/bindings/employees-accesspoints?employeeId=1&employeeId=2) ▪ accessPointId - identifier(s) of access point(s) - returns only records with specified identifiers (e.g. api/v1/bindings/employees-accesspoints?accessPointId=1) ▪ limit - specifies how many entities the response should contain (e.g. api/v1/bindings/employees-accesspoints?limit=10 - means that only 10 entities returned in a response). The value of this parameter must be between 0 and 3000 ▪ offset - specifies how many entities should be skipped from the top (e.g. api/v1/bindings/employees-accesspoints?offset=10 - means that 10 records are skipped and next 10 entities returned in a response). The value of this parameter must be greater than or equal to 0
Body	
Success Response	Status code: 200 Content-type: application/json Body: <pre>[{ "employeeId": <Number>, "accessPointId": <Number> }, ...]</pre>
Error Response	<u>401 (Unauthorized)</u> <u>403 (Permission denied)</u> <u>500 (Internal Server Error)</u>
Example	Request: <pre>curl -X GET http://192.168.0.108:9500/api/v1/bindings/employees-accesspoints -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJ"</pre> Response: <pre>[{ "employeeId": 7, "accessPointId": 1 }, { "employeeId": 8, "accessPointId": 2 }]</pre>

Create bindings between Employees and Access Points

URL Endpoint	api/v1/bindings/employees-accesspoints
Method	POST
Header Parameters	Authorization: Bearer <JWT access token> Content-Type: application/json
Path Parameters	
Query Parameters	
Body	<pre>[{ "employeeIds": [<Number>], "accessPointIds": [<Number>] }]</pre> Required fields: - employeeIds - accessPointIds
Success Response	Status code: 204 No Content
Error Response	401 (Unauthorized) 403 (Permission denied) 422 (Validation Error) 500 (Internal Server Error)
Example	Request: <pre>curl -X POST http://192.168.0.108:9500/api/v1/bindings/employees-accesspoints -H "Content-Type: application/json" -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJ" -d '[{ "employeeIds": [7,8], "accessPointIds": [3] }]</pre> Response: 204 No Content

Delete bindings between Employees and Access Points

URL Endpoint	api/v1/bindings/employees-accesspoints/delete
Method	POST
Header Parameters	Authorization: Bearer <JWT access token> Content-Type: application/json
Path Parameters	
Query Parameters	
Body	<pre>[{ "employeeIds": [<Number>], "accessPointIds": [<Number>] }]</pre> Required fields: - employeeIds - accessPointIds
Success Response	Status code: 204 No Content
Error Response	401 (Unauthorized) 403 (Permission denied) 422 (Validation Error) 500 (Internal Server Error)
Example	Request: <pre>curl -X POST http://192.168.0.108:9500/api/v1/bindings/employees-accesspoints/delete -H "Content-Type: application/json" -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJ" -d '[{ "employeeIds": [7,8], "accessPointIds": [2,3] }]</pre> Response: 204 No Content

2.8.5. Official Vehicles-Cards

Get Official Vehicles Cards

URL Endpoint	api/v1/bindings/vehicles-cards
Method	GET
Header Parameters	Authorization: Bearer <JWT access token>
Path Parameters	
Query Parameters	vehicleId - identifier(s) of Official Vehicle(s) - returns only records with specified identifiers (e.g. api/v1/bindings/vehicles-cards?vehicleId=1,2 or api/v1/bindings/vehicles-cards?vehicleId=1&vehicleId=2) cardId - IDs of Card(s) - returns only records with specified identifiers (e.g. api/v1/bindings/vehicles-cards?cardId=1,2 or api/v1/bindings/vehicles-cards?cardId=1&cardId=2) limit - specifies how many entities the response should contain (e.g. api/v1/bindings/vehicles-cards?limit=10 - means that only 10 entities returned in a response). The value of this parameter must be between 0 and 3000 offset - specifies how many entities should be skipped from the top (e.g. api/v1/bindings/vehicles-cards?limit=10&offset=10 - means that 10 records are skipped and next 10 entities returned in a response). The value of this parameter must be greater than or equal to 0
Body	
Success Response	Status code: 200 Content-type: application/json Body: <pre>[{ "vehicleId": <Number>, "cardId": <Number>, "startDate": <Date>,// card validity start "expirationDate": <Date>,// card validity end "format": <Enum: [W26, W34, W36, W37, W42, W58, W58DEC]> }, ...]</pre>
Error Response	<u>400 (Invalid request)</u> <u>401 (Unauthorized)</u> <u>403 (Permission denied)</u> <u>500 (Internal Server Error)</u>
Example	Request: <pre>curl -X GET http://192.168.0.108:9500/api/v1/bindings/vehicles-cards -H "Content-Type: application/json" -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilslm"</pre> Response: <pre>[{ "vehicleId": 1, "cardId": 1, "startDate": "2023-12-03T21:41:45.384Z", "expirationDate": "2023-12-03T21:41:45.384Z", "format": "W26" }]</pre>

Update expiration time for Cards of Official Vehicles

URL Endpoint	api/v1/bindings/vehicles-cards
Method	PUT
Header Parameters	Authorization: Bearer <JWT access token> Content-Type: application/json
Path Parameters	
Query Parameters	
Body	<pre>[{ "vehicleId": <Number>, "cardId": <Number>, "startDate": <Date>, "expirationDate": <Date>, "format": <Enum: [W26, W34, W36, W37, W42, W58, W58DEC]> }, ...]</pre> Required fields: - vehicleId - cardId - expirationDate
Success Response	Status code: 204 No Content
Error Response	400 (Invalid request) 401 (Unauthorized) 403 (Permission denied) 422 (Validation Error) 500 (Internal Server Error)
Example	Request: <pre>curl -X PUT http://192.168.0.108:9500/api/v1/bindings/vehicles-cards -H "Content-Type: application/json" -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilsm" -d '{"vehicleId":1, "cardId": 2, "format": "W26", "startDate": "2023-12-01T21:45:01.520Z", "expirationDate": "2023-12-03T21:45:01.520Z"}'</pre> Response: 204 No Content

Assign Cards to Official Vehicles

URL Endpoint	api/v1/bindings/vehicles-cards
Method	POST
Header Parameters	Authorization: Bearer <JWT access token> Content-Type: application/json
Path Parameters	
Query Parameters	
Body	<pre>[{ "vehicleId": <Number>, "cardId": <Number>, "startDate": <Date>, "expirationDate": <Date>, "format": <Enum: [W26, W34, W36, W37, W42, W58, W58DEC]> }, ...]</pre> Required fields: • vehicleId • cardId • format
Success Response	Status code: 204 No Content
Error Response	400 (Invalid request) 401 (Unauthorized) 403 (Permission denied) 422 (Validation Error) 500 (Internal Server Error)
Example	Request: <pre>curl -X POST http://192.168.0.108:9500/api/v1/bindings/vehicles-cards -H "Content-Type: application/json" -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilsm" -d '{"vehicleId":1,"cardId": 2,"format": "W26", "startDate": "2023-12-01T21:48:38.836Z","expirationDate": "2023-12-03T21:48:38.836Z"}'</pre> Response: 204 No Content

Remove Cards from Official Vehicles

URL Endpoint	api/v1/bindings/vehicles-cards/delete
Method	POST
Header Parameters	Authorization: Bearer <JWT access token> Content-Type: application/json
Path Parameters	
Query Parameters	
Body	<pre>[{ "vehicleId": <Number>, "cardId": <Number>, "startDate": <Date>, "expirationDate": <Date>, "format": <Enum: [W26, W34, W36, W37, W42, W58, W58DEC]> }, ...]</pre> Required fields: - vehicleId - cardId - format
Success Response	Status code: 204 No Content
Error Response	400 (Invalid request) 401 (Unauthorized) 403 (Permission denied) 422 (Validation Error) 500 (Internal Server Error)
Example	Request: <pre>curl -X POST http://192.168.0.108:9500/api/v1/bindings/vehicles-cards/delete -H "Content-Type: application/json" -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilsm" -d '{"vehicleId":1, "cardId": 2, "format": "W26", "startDate": "2023-12-01T21:57:58.357Z", "expirationDate": "2023-12-03T21:57:58.357Z"}'</pre> Response: 204 No Content

2.8.6. Official Vehicles-Access Rules

Get Official Vehicles Access Rules

URL Endpoint	api/v1/bindings/vehicles-accessrules
Method	GET
Header Parameters	Authorization: Bearer <JWT access token>
Path Parameters	
Query Parameters	▪ vehicleId - identifier(s) of Official Vehicle(s) - returns only records with specified identifiers (e.g. api/v1/bindings/vehicles-accessrules?vehicleId=1,2 or api/v1/bindings/vehicles-accessrules?vehicleId=1&vehicleId=2) ▪ accessruleId - IDs of Acces Rule (s) - returns only records with specified identifiers (e.g. api/v1/binding/vehicles-accessrules?accessrulesId=1,2 or api/v1/bindings/vehicles-accessrules?accessruleId=1&accessrulesId=2) ▪ limit - specifies how many entities the response should contain (e.g. api/v1/bindings/vehicles-accessrules?limit=10 - means that only 10 entities returned in a response). The value of this parameter must be between 0 and 3000 ▪ offset - specifies how many entities should be skipped from the top (e.g. api/v1/bindings/vehicles-accessrules?limit=10&offset=10 - means that 10 records are skipped and next 10 entities returned in a response). The value of this parameter must be greater than or equal to 0
Body	
Success Response	Status code: 200 Content-type: application/json Body: <pre>[{ "vehicleId": <Number>, "accessruleId": <Number> }, ...]</pre>
Error Response	<u>400 (Invalid request)</u> <u>401 (Unauthorized)</u> <u>403 (Permission denied)</u> <u>500 (Internal Server Error)</u>
Example	Request: <pre>curl -X GET http://192.168.0.108:9500/api/v1/bindings/vehicles-accessrules -H "Content-Type: application/json" -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiIjojVXNlcilsm"</pre> Response: <pre>[{ "vehicleId": 1, "accessruleId": 1 }, { "vehicleId": 1, "accessruleId": 2 }, { "vehicleId": 2, "accessruleId": 1 }]</pre>

Assign Access Rules to Official Vehicle

URL Endpoint	api/v1/bindings/vehicles-accessrules
Method	POST
Header Parameters	Authorization: Bearer <JWT access token> Content-Type: application/json
Path Parameters	
Query Parameters	
Body	<pre>[{ "vehicleId": <Number>, "accessruleId": <Number> }, ...]</pre> Required fields: - vehicleId - accessruleId
Success Response	Status code: 204 No Content
Error Response	400 (Invalid request) 401 (Unauthorized) 403 (Permission denied) 422 (Validation Error) 500 (Internal Server Error)
Example	Request: <pre>curl -X POST http://192.168.0.108:9500/api/v1/bindings/vehicles-accessrules -H "Content-Type: application/json" -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilsm" -d '{"vehicleId":1,"accessruleId":1}, {"vehicleId":2, "accessruleId":2}]'</pre> Response: 204 No Content

Remove Access Rules from Official Vehicles

URL Endpoint	api/v1/bindings/vehicles-accessrules/delete
Method	POST
Header Parameters	Authorization: Bearer <JWT access token> Content-Type: application/json
Path Parameters	
Query Parameters	
Body	<pre>[{ "vehicleId": <Number>, "accessruleId": <Number> }, ...]</pre> Required fields: - vehicleId - accessruleId
Success Response	Status code: 204 No Content
Error Response	400 (Invalid request) 401 (Unauthorized) 403 (Permission denied) 422 (Validation Error) 500 (Internal Server Error)
Example	Request: <pre>curl -X POST http://192.168.0.108:9500/api/v1/bindings/vehicles-accessrules/delete -H "Content-Type: application/json" -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilsm" -d '{"vehicleId":1,"accessruleId":1}, {"vehicleId":2, "accessruleId":2}]'</pre> Response: 204 No Content

2.8.7. Official Vehicles-Access Points

Get bindings between Official Vehicles and Access Points

URL Endpoint	api/v1/bindings/vehicles-accesspoints
Method	GET
Header Parameters	Authorization: Bearer <JWT access token>
Path Parameters	
Query Parameters	▪ vehicleIds - identifier(s) of Official Vehicle(s) - returns only records with specified identifiers (e.g. api/v1/bindings/vehicles-accesspoints?vehicleIds=1,2 or api/v1/bindings/vehicles-accesspoints?vehicleIds=1&vehicleIds=2) ▪ accessPointId - identifier(s) of access point(s) - returns only records with specified identifiers (e.g. api/v1/bindings/vehicles-accesspoints?accessPointId=1) ▪ limit - specifies how many entities the response should contain (e.g. api/v1/bindings/vehicles-accesspoints?limit=10 - means that only 10 entities returned in a response). The value of this parameter must be between 0 and 3000 ▪ offset - specifies how many entities should be skipped from the top (e.g. api/v1/bindings/vehicles-accesspoints?offset=10 - means that 10 records are skipped and next 10 entities returned in a response). The value of this parameter must be greater than or equal to 0
Body	
Success Response	Status code: 200 Content-type: application/json Body: <pre>[{ "vehicleId": <Number>, "accessPointId": <Number> }, ...]</pre>
Error Response	<u>401 (Unauthorized)</u> <u>403 (Permission denied)</u> <u>500 (Internal Server Error)</u>
Example	Request: <pre>curl -X GET http://192.168.0.108:9500/api/v1/bindings/vehicles-accesspoints -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJ"</pre> Response: <pre>[{ "vehicleId": 10, "accessPointId": 1 }, { "vehicleId": 10, "accessPointId": 2 }]</pre>

Create bindings between Vehicles and Access Points

URL Endpoint	api/v1/bindings/vehicles-accesspoints
Method	POST
Header Parameters	Authorization: Bearer <JWT access token> Content-Type: application/json
Path Parameters	
Query Parameters	
Body	<pre>[{ "vehicleIds": [<Number>], "accessPointIds": [<Number>] }]</pre> Required fields: - vehicleIds - accessPointIds
Success Response	Status code: 204 No Content
Error Response	401 (Unauthorized) 403 (Permission denied) 422 (Validation Error) 500 (Internal Server Error)
Example	Request: <pre>curl -X POST http://192.168.0.108:9500/api/v1/bindings/vehicles-accesspoints -H "Content-Type: application/json" -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJ" -d '[{ "vehicleIds": [10], "accessPointIds": [2,3] }]</pre> Response: 204 No Content

Delete bindings between Vehicles and Access Points

URL Endpoint	api/v1/bindings/vehicles-accesspoints/delete
Method	POST
Header Parameters	Authorization: Bearer <JWT access token> Content-Type: application/json
Path Parameters	
Query Parameters	
Body	<pre>[{ "vehicleIds": [<Number>], "accessPointIds": [<Number>] }]</pre> Required fields: - vehicleIds - accessPointIds
Success Response	Status code: 204 No Content
Error Response	401 (Unauthorized) 403 (Permission denied) 422 (Validation Error) 500 (Internal Server Error)
Example	Request: <pre>curl -X POST http://192.168.0.108:9500/api/v1/bindings/vehicles-accesspoints/delete -H "Content-Type: application/json" -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJ" -d '[{ "vehicleIds": [10], "accessPointIds": [1] }]</pre> Response: 204 No Content

2.8.8. Access Rules-Access Points

Get bindings between Access Rules and Access Points

URL Endpoint	api/v1/bindings/accessrules-accesspoints
Method	GET
Header Parameters	Authorization: Bearer <JWT access token>
Path Parameters	
Query Parameters	accessRuleId - identifier(s) of access rule(s) - returns only records with specified identifiers (e.g. api/v1/bindings/accessrules-accesspoints?accessRuleId=6) accessPointId - identifier(s) of access point(s) - returns only records with specified identifiers (e.g. api/v1/bindings/accessrules-accesspoints?accessPointId=1) limit - specifies how many entities the response should contain (e.g. api/v1/bindings/accessrules-accesspoints?limit=10 - means that only 10 entities returned in a response). The value of this parameter must be between 0 and 3000 offset - specifies how many entities should be skipped from the top (e.g. api/v1/bindings/accessrules-accesspoints?offset=10 - means that 10 records are skipped and next 10 entities returned in a response). The value of this parameter must be greater than or equal to 0
Body	
Success Response	Status code: 200 Content-type: application/json Body: <pre>[{ "accessRuleId": <Number>, "accessPointId": <Number> }, ...]</pre>
Error Response	400 (Invalid request) 401 (Unauthorized) 403 (Permission denied) 500 (Internal Server Error)
Example	Request: <pre>curl -X GET http://192.168.0.108:9500/api/v1/bindings/accessrules-accesspoints -H "Content-Type: application/json" -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilslm"</pre> Response: <pre>[{ "accessRuleId": 6, "accessPointId": 1 }, { "accessRuleId": 4, "accessPointId": 7 },]</pre>

Assign Access Aoints to Access Rules

URL Endpoint	api/v1/bindings/accessrules-accesspoints
Method	POST
Header Parameters	Authorization: Bearer <JWT access token> Content-Type: application/json
Path Parameters	
Query Parameters	
Body	<pre>[{ "accessruleId": <Number>, "accesspointId": <Number>, }, ...]</pre> Required fields: - accessruleId - accesspointId
Success Response	Status code: 204 No Content
Error Response	400 (Invalid request) 401 (Unauthorized) 403 (Permission denied) 422 (Validation Error) 500 (Internal Server Error)
Example	Request: <pre>curl -X POST http://192.168.0.108:9500/api/v1/bindings/accessrules-accesspoints -H "Content-Type: application/json" -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilsm" -d '[{ "accessruleId":6, "accesspointId":2}]'</pre> Response: 204 No Content

Unassign Access Points to Access Rules

URL Endpoint	api/v1/bindings/accessrules-accesspoints/delete
Method	POST
Header Parameters	Authorization: Bearer <JWT access token> Content-Type: application/json
Path Parameters	
Query Parameters	
Body	<pre>[{ "accessruleId": <Number>, "accesspointId": <Number>, }, ...]</pre> Required fields: - accessruleId - accesspointId
Success Response	Status code: 204 No Content
Error Response	400 (Invalid request) 401 (Unauthorized) 403 (Permission denied) 422 (Validation Error) 500 (Internal Server Error)
Example	Request: <pre>curl -X POST http://192.168.0.108:9500/api/v1/bindings/accessrules-accesspoints/delete -H "Content-Type: application/json" -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlciIsIm" -d '[{ "accessruleId":6, "accesspointId":2}]'</pre> Response: 204 No Content

2.8.9. Guests-Access Rules

Get Guests Access Rules

URL Endpoint	api/v1/bindings/guests-accessrules
Method	GET
Header Parameters	Authorization: Bearer <JWT access token>
Path Parameters	
Query Parameters	guestId — identifier(s) of Guest(s). Returns only records with the specified identifiers (e.g. api/v1/bindings/guests-accessrules?guestId=1,2 or api/v1/bindings/guests-accessrules?guestId=1&guestId=2) accessRuleId — identifier(s) of Access Rule(s). Returns only records with the specified identifiers (e.g. api/v1/binding/guests-accessrules?accessRuleId=1,2 or api/v1/bindings/guests-accessrules?accessRuleId=1&accessRuleId=2) limit — specifies how many entities the response should contain (e.g. api/v1/bindings/guests-accessrules?limit=10 — means that only 10 entities are returned). The value of this parameter must be between 0 and 3000 offset — specifies how many entities should be skipped from the top (e.g. api/v1/bindings/guests-accessrules?limit=10&offset=10 — means that 10 records are skipped, and next 10 entities are returned). The value of this parameter must be greater than or equal to 0
Body	
Success Response	Status code: 200 Content-type: application/json Body: <pre>[{ "guestId": <Number>, "accessRuleId": <Number>, }, ...]</pre>
Error Response	400 (Invalid Request) 401 (Unauthorized) 403 (Permission Denied) 500 (Internal Server Error)
Example	Request: <pre>curl -X GET http://192.168.0.108:9500/api/v1/bindings/guests-accessrules -H "Content-Type: application/json" -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilsm"</pre> Response: <pre>[{"guestId": 1, "accessRuleId": 1}, {"guestId": 1, "accessRuleId": 2}, {"guestId": 2, "accessRuleId": 1}]</pre>

Assign Access Rules to Guests

URL Endpoint	/api/v1/bindings/guests-accessrules
Method	POST
Header Parameters	Authorization: Bearer <JWT access token> Content-Type: application/json
Path Parameters	
Query Parameters	
Body	<pre>[{ "guestId": <Number>, "accessRuleId ": <Number> }, ...]</pre> Required fields: • guestId • accessRuleId The rule type must be NORMAL, and the option "The rule can be accessed from the Guests tab" must be enabled in the "Client" tool under the "Rules" section.
Success Response	Status code: 204 No Content
Error Response	<u>400 (Invalid Request)</u> <u>401 (Unauthorized)</u> <u>403 (Permission Denied)</u> <u>422 (Validation Error)</u> <u>500 (Internal Server Error)</u>
Example	Request: <pre>curl -X POST http://192.168.0.108:9500/api/v1/bindings/guests-accessrules -H "Content-Type: application/json" -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilsm" -d '[{ "guestId":1, "accessRuleId":1}, {"guestId":2, "accessRuleId":2}]'</pre> Response: 204 No Content

Unassign Access Rules from Guests

URL Endpoint	api/v1/bindings/guests-accessrules/delete
Method	POST
Header Parameters	Authorization: Bearer <JWT access token> Content-Type: application/json
Path Parameters	
Query Parameters	
Body	<pre>[{ "guestId": <Number>, "accessRuleId ": <Number> }, ...]</pre> Required fields: • guestId • accessRuleId
Success Response	Status code: 204 No Content
Error Response	400 (Invalid Request) 401 (Unauthorized) 403 (Permission Denied) 422 (Validation Error) 500 (Internal Server Error)
Example	Request: <pre>curl -X POST http://192.168.0.108:9500/api/v1/bindings/guests-accessrules/delete -H "Content-Type: application/json" -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilslm" -d '[{ "guestId":1, "accessRuleId":1}, {"guestId":2, "accessRuleId":2}]'</pre> Response: 204 No Content

2.9. Access rules

2.9.1. Get all access rules

URL Endpoint	api/v1/accessrules
Method	GET
Header Parameters	Authorization: Bearer <JWT access token>
Path Parameters	
Query Parameters	▪ id — access rule identifier(s). Returns only records with the specified identifiers (e.g. api/v1/accessrules?id=1,2 or api/v1/accessrules?id=1&id=2) ▪ level — filter by access rule level (e.g. api/v1/accessrules?level=2) ▪ name — filter by access rule name with condition "starts with" (e.g. api/v1/accessrules?name=Rule1) ▪ limit — specifies how many entities the response should contain (e.g. api/v1/accessrules?limit=10 — means that only 10 entities are returned). The value of this parameter must be between 0 and 3000 ▪ offset — specifies how many entities should be skipped from the top (e.g. api/v1/accessrules?limit=10&offset=10 — means that 10 records are skipped, and next 10 entities are returned). The value of this parameter must be greater than or equal to 0 ▪ includeFields — specifies what fields the response should contain (e.g. api/v1/accessrules?includeFields=name — only the "name" field is returned) ▪ excludeFields — specifies what fields should be omitted from the response (e.g. api/v1/accessrules?excludeFields=id — all fields except "id" are returned) ▪ sortBy — specifies the sort field (e.g. api/v1/accessrules?sortBy=name — means that entities are sorted by the "name" field) ▪ sortOrder — specifies the sort order. Can be either ASC or DESC (e.g. api/v1/accessrules?sortBy=name&sortOrder=DESC — means that entities are sorted by the "name" field in descending order) ▪ folderId — filter by folder ID (e.g. api/v1/accessrules?folderId=2) ▪ type — filter by access rule type. Possible values: NORMAL, DOORS (e.g. api/v1/accessrules?type=NORMAL) ▪ startDate[operation] — filter by start date of access rule. Possible values for [operation]: EQ, GT, LT, GTE, LTE (e.g. api/v1/accessrules?startDate%5BGTE%5D=2023-05-10) ▪ endDate[operation] — filter by end date of access rule. Possible values for [operation]: EQ, GT, LT, GTE, LTE (e.g. api/v1/accessrules?endDate%5BGTE%5D=2023-05-10) ▪ isGlobal — specifies whether to return only global or only local access rules. All access rules are returned if not specified (default) (e.g. api/v1/accessrules?isGlobal=true) ▪ includeIntervals — specifies whether to return the full day-by-day interval schedule per rule. No intervals are returned if not specified (false by default) (e.g. api/v1/accessrules?includeIntervals=true)
Body	

Success Response	Status code: 200 Content-type: application/json Body: <pre>[{ "id": <Number>, // access rule identifier "name": <String>, // access rule name "description": <String>, // access rule description "startDate": <Date>, // access rule start date "endDate": <Date>, // access rule end date "level": <Number>, // access rule priority level (1-4) "folderId": <Number>, // access rule folder ID "isGlobal": <Boolean>, // indicates whether the access rule is global or local "intervals": { // all access intervals for the access rule "cyclicDays": [// access intervals for cyclic days (a repeating sequence of days until the rule expires) { "accessIn": [// entry intervals { "start": <Time>, // access is allowed from this time "end": <Time> // access is allowed until this time }], "accessOut": [// exit intervals { "start": <Time>, // access is allowed from this time "end": <Time> // access is allowed until this time }] }, ...], "standardWeek": <Boolean>, // when true, the Client tool shows weekday names instead of day numbers "specialDays": [// access intervals for specific dates { "date": <Date>, // specific date "accessIn": [// entry intervals { "start": <Time>, // access is allowed from this time "end": <Time> // access is allowed until this time }], "accessOut": [// exit intervals { "start": <Time>, // access is allowed from this time "end": <Time> // access is allowed until this time }] }, ...] }, ...], ...]</pre>
Error Response	<u>400 (Invalid Request)</u> <u>401 (Unauthorized)</u> <u>403 (Access Denied)</u> <u>500 (Internal Server Error)</u>

	Request: <pre>curl -X 'GET' \ 'http://192.168.0.108:9500/api/v1/accessrules?includeIntervals=true' \ -H 'Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ0eXNlbnRlcilslm'</pre> Response: <pre>[{ "id": 1, "name": "Default", "description": "No access", "startDate": "2001-01-01", "endDate": "2026-04-02", "level": 1, "folderId": 1, "isGlobal": false }, { "id": 5, "name": "1/1", "description": "1 day of work and 1 day off", "startDate": "2026-04-03", "endDate": "2027-04-03", "level": 2, "folderId": 9, "isGlobal": false, "intervals": { "cyclicDays": [{ "accessIn": [{"start": "06:00", "end": "22:00"}], "accessOut": [{"start": "06:00", "end": "22:00"}] }, {}], "standardWeek": false, "specialDays": [{ "date": "2026-10-31", "accessIn": [{"start": "08:00", "end": "20:00"}], "accessOut": [{"start": "08:00", "end": "20:00"}] }] } }]</pre>
--	--

Note: Empty "accessIn" / "accessOut" arrays are omitted from the response (see [Create access rule](#)).

2.9.2. Get access rule by ID

URL Endpoint	api/v1/accessrules/{id}
Method	GET
Header Parameters	Authorization: Bearer <JWT access token>
Path Parameters	id — access rule identifier
Query Parameters	includeFields — specifies what fields the response should contain (e.g. api/v1/accessrules/1?includeFields=name — only the "name" field is returned) excludeFields — specifies what fields should be omitted from the response (e.g. api/v1/accessrules/1?excludeFields=id — all fields except "id" are returned) includeIntervals — specifies whether to return the full day-by-day interval schedule. No intervals are returned if not specified (false by default) (e.g. api/v1/accessrules/1?includeIntervals=true)
Body	
Success Response	Status code: 200 Content-type: application/json Body: <pre>{ "id": <Number>, // access rule identifier "name": <String>, // access rule name "description": <String>, // access rule description "startDate": <Date>, // access rule start date "endDate": <Date>, // access rule end date "level": <Number>, // access rule priority level (1-4) "folderId": <Number>, // access rule folder ID "isGlobal": <Boolean>, // indicates whether the access rule is global or local "intervals": { // all access intervals for the access rule "cyclicDays": [// access intervals for cyclic days (a repeating sequence of days until the rule expires) { "accessIn": [// entry intervals { "start": <Time>, // access is allowed from this time "end": <Time> // access is allowed until this time }], "accessOut": [// exit intervals { "start": <Time>, // access is allowed from this time "end": <Time> // access is allowed until this time }] }, ...], "standardWeek": <Boolean>, // when true, the Client tool shows weekday names instead of day numbers "specialDays": [// access intervals for specific dates { "date": <Date>, // specific date "accessIn": [// entry intervals { "start": <Time>, // access is allowed from this time "end": <Time> // access is allowed until this time }], "accessOut": [// exit intervals { "start": <Time>, // access is allowed from this time "end": <Time> // access is allowed until this time }] }, ...] } }</pre>

Error Response	400 (Invalid Request) 401 (Unauthorized) 403 (Access Denied) 404 (Not Found) 500 (Internal Server Error)
Example	Request: <pre>curl -X 'GET' \ 'http://192.168.0.108:9500/api/v1/accessrules/5?includeIntervals=true' \ -H 'Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlciIsIm'</pre> Response: <pre>{ "id": 5, "name": "1/1", "description": "1 day of work and 1 day off", "startDate": "2026-04-03", "endDate": "2027-04-03", "level": 2, "folderId": 9, "isGlobal": false, "intervals": { "cyclicDays": [{ "accessIn": [{"start": "06:00", "end": "22:00"}], "accessOut": [{"start": "06:00", "end": "22:00"}] }, {}], "standardWeek": false, "specialDays": [{ "date": "2026-10-31", "accessIn": [{"start": "08:00", "end": "20:00"}], "accessOut": [{"start": "08:00", "end": "20:00"}] }] } }</pre> Note: Empty "accessIn" / "accessOut" arrays are omitted from the response (see Create access rule).

2.9.3. Create access rule

URL Endpoint	api/v1/accessrules
Method	POST
Header Parameters	Authorization: Bearer <JWT access token> Content-Type: application/json
Path Parameters	
Query Parameters	
Body	<pre>{ "name": <String>, "description": <String>, "startDate": <Date>, "endDate": <Date>, "level": <Number>, "folderId": <Number>, "intervals": { "cyclicDays": [{ "accessIn": [{"start": <Time>, "end": <Time>}], "accessOut": [{"start": <Time>, "end": <Time>}] }, ...], "standardWeek": <Boolean>, "specialDays": [{ "date": <Date>, "accessIn": [{"start": <Time>, "end": <Time>}], "accessOut": [{"start": <Time>, "end": <Time>}] }, ...] } }</pre> Required fields: - name - startDate (YYYY-MM-DD) - endDate (YYYY-MM-DD) - level (1–4, where 1 is the lowest and 4 is the highest) Optional fields: - description - folderId - standardWeek — when true, the Client tool labels cyclic days with weekday names ("Mon.", "Tues.", and so on) instead of day numbers. Applies only when the number of cyclic days is a multiple of seven. Affects display only and does not change access behavior. Default: false - intervals — if omitted, the access rule has no days or time intervals and therefore grants no access. If provided: - Overlapping time intervals within a single day are automatically merged. - Special days are sorted chronologically in the response. - Empty "accessIn" / "accessOut" arrays are omitted from the response (see response example below). - A maximum of 32 cyclic days and 32 special days is allowed. "start" / "end" format: HH:MM. "date" format: YYYY-MM-DD. See examples below.

Body	To deny access on a specific day, you must send empty arrays for "accessIn" and "accessOut" for that day. Example: <pre> "intervals": { "cyclicDays": [{ "accessIn": [{"start": "06:00", "end": "22:00"}], "accessOut": [{"start": "06:00", "end": "22:00"}] }, { "accessIn": [], "accessOut": [] }] } </pre> In this example: day 1 = access 06:00–22:00, day 2 = no access. This 2-day cycle repeats until the access rule expires. To completely deny access, you must specify a single item in the "cyclicDays" array with empty "accessIn" and "accessOut" arrays. Example: <pre> "intervals": { "cyclicDays": [{ "accessIn": [], "accessOut": [] }] } </pre>
Success Response	Status code: 200 Content-type: application/json Body: <pre> { "id": <Number>, "name": <String>, "description": <String>, "startDate": <Date>, "endDate": <Date>, "level": <Number>, "folderId": <Number>, "isGlobal": <Boolean>, "intervals": { "cyclicDays": [{ "accessIn": [{"start": <Time>, "end": <Time>}], "accessOut": [{"start": <Time>, "end": <Time>}] }, ...], "standardWeek": <Boolean>, "specialDays": [{ "date": <Date>, "accessIn": [{"start": <Time>, "end": <Time>}], "accessOut": [{"start": <Time>, "end": <Time>}] }, ...] } } </pre>
Error Response	400 (Invalid Request) 401 (Unauthorized) 403 (Access Denied) 404 (Not Found) 422 (Validation Error) 500 (Internal Server Error)

Example	Request: <pre>curl -X 'POST' \ 'http://192.168.0.108:9500/api/v1/accessrules' \ -H 'Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilslm' \ -H 'Content-Type: application/json' \ -d '{\"name\":\"1/1\",\"description\":\"1 day of work and 1 day off\",\"startDate\":\"2026-04-03\",\"endDate\":\"2027-04-03\", \"level\":2,\"folderId\":9,\"intervals\":{\"cyclicDays\":{\"accessIn\":{\"start\":\"06:00\",\"end\":\"22:00\"}, \"accessOut\":{\"start\":\"06:00\",\"end\":\"22:00\"}}, {\"accessIn\":[],\"accessOut\":[]}}, \"specialDays\":{\"date\":\"2026-10-31\",\"accessIn\":{\"start\":\"08:00\",\"end\":\"20:00\"}, \"accessOut\":{\"start\":\"08:00\",\"end\":\"20:00\"}}}}'</pre> Response: <pre>{ \"id\": 5, \"name\": \"1/1\", \"description\": \"1 day of work and 1 day off\", \"startDate\": \"2026-04-03\", \"endDate\": \"2027-04-03\", \"level\": 2, \"folderId\": 9, \"isGlobal\": false, \"intervals\": { \"cyclicDays\": [{ \"accessIn\": [{\"start\": \"06:00\", \"end\": \"22:00\"}], \"accessOut\": [{\"start\": \"06:00\", \"end\": \"22:00\"}] }, {}], \"standardWeek\": false, \"specialDays\": [{ \"date\": \"2026-10-31\", \"accessIn\": [{\"start\": \"08:00\", \"end\": \"20:00\"}], \"accessOut\": [{\"start\": \"08:00\", \"end\": \"20:00\"}] }] } }</pre> Note: Empty "accessIn" / "accessOut" arrays are omitted from the response.
---------	---

2.9.4. Update access rule

URL Endpoint	api/v1/accessrules/{id}
Method	PUT
Header Parameters	Authorization: Bearer <JWT access token> Content-Type: application/json
Path Parameters	id — access rule identifier
Query Parameters	
Body	<pre> { "name": <String>, "description": <String>, "startDate": <Date>, "endDate": <Date>, "level": <Number>, "folderId": <Number>, "intervals": { "cyclicDays": [{ "accessIn": [{"start": <Time>, "end": <Time>}], "accessOut": [{"start": <Time>, "end": <Time>}], }, ...], "standardWeek": <Boolean>, "specialDays": [{ "date": <Date>, "accessIn": [{"start": <Time>, "end": <Time>}], "accessOut": [{"start": <Time>, "end": <Time>}], }, ...] } } </pre> Note: Only the fields included in the request body are updated. Other fields remain unchanged. The same applies to the "intervals" attribute: - If omitted, the day and time access intervals remain unchanged. - If provided, the existing configuration is replaced with the new one. For rules and examples of how to specify access intervals, see Create access rule.

Success Response	Status code: 200 Content-type: application/json Body: <pre>{ "id": <Number>, "name": <String>, "description": <String>, "startDate": <Date>, "endDate": <Date>, "level": <Number>, "folderId": <Number>, "isGlobal": <Boolean>, "intervals": { "cyclicDays": [{ "accessIn": [{"start": <Time>, "end": <Time>}], "accessOut": [{"start": <Time>, "end": <Time>}] }, ...], "standardWeek": <Boolean>, "specialDays": [{ "date": <Date>, "accessIn": [{"start": <Time>, "end": <Time>}], "accessOut": [{"start": <Time>, "end": <Time>}] }, ...] } }</pre>
Error Response	400 (Invalid Request) 401 (Unauthorized) 403 (Access Denied) 404 (Not Found) 422 (Validation Error) 500 (Internal Server Error)

Example	Request: <pre>curl -X 'PUT' \ 'http://192.168.0.108:9500/api/v1/accessrules/5' \ -H 'Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilslm' \ -H 'Content-Type: application/json' \ -d '{"endDate": "2028-04-03"}'</pre> Response: <pre>{ "id": 5, "name": "1/1", "description": "1 day of work and 1 day off", "startDate": "2026-04-03", "endDate": "2028-04-03", "level": 2, "folderId": 9, "isGlobal": false, "intervals": { "cyclicDays": [{ "accessIn": [{"start": "06:00", "end": "22:00"}], "accessOut": [{"start": "06:00", "end": "22:00"}] }, {}], "standardWeek": false, "specialDays": [{ "date": "2026-10-31", "accessIn": [{"start": "08:00", "end": "20:00"}], "accessOut": [{"start": "08:00", "end": "20:00"}] }] } }</pre>
---------	---

2.9.5. Delete access rule

URL Endpoint	api/v1/accessrules/{id}
Method	DELETE
Header Parameters	Authorization: Bearer <JWT access token>
Path Parameters	id – access rule identifier
Query Parameters	
Body	
Success Response	Status code: 200 Content-type: application/json Body: <pre>{ "id": <Number>, "name": <String>, "description": <String>, "startDate": <Date>, "endDate": <Date>, "level": <Number>, "folderId": <Number>, "isGlobal": <Boolean> }</pre>
Error Response	400 (Invalid Request) 401 (Unauthorized) 403 (Access Denied) 404 (Not Found) 422 (Validation Error) 500 (Internal Server Error)
Example	Request: <pre>curl -X 'DELETE' \ 'http://192.168.0.108:9500/api/v1/accessrules/5' \ -H 'Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlciIsIm'</pre> Response: <pre>{ "id": 5, "name": "1/1", "description": "1 day of work and 1 day off", "startDate": "2026-04-03", "endDate": "2028-04-03", "level": 2, "folderId": 9, "isGlobal": false }</pre>

2.9.6. Count access rules

URL Endpoint	api/v1/accessrules/count
Method	GET
Header Parameters	Authorization: Bearer <JWT access token>
Path Parameters	
Query Parameters	id — access rule identifier(s). Returns the number of records with the specified identifiers (e.g. api/v1/accessrules/count?id=6&id=1) level — filter by access rule level (e.g. api/v1/accessrules/count?level=1) name — filter by access rule name with condition "starts with" (e.g. api/v1/accessrules/count?name=free) folderId — filter by folder ID (e.g. api/v1/accessrules/count?folderId=2) type — filter by access rule type. Possible values: NORMAL, DOORS (e.g. api/v1/accessrules/count?type=NORMAL) startDate[operation] — filter by start date of access rule. Possible values for [operation]: EQ, GT, LT, GTE, LTE (e.g. api/v1/accessrules/count?startDate%5BGTE%5D=2023-05-10) endDate[operation] — filter by end date of access rule. Possible values for [operation]: EQ, GT, LT, GTE, LTE (e.g. api/v1/accessrules/count?endDate%5BGTE%5D=2023-05-10) groupBy — specifies which field to use for grouping. With groupBy set, the result is a list of per-group counts instead of a single total. Possible values: name, level, folderId (e.g. api/v1/accessrules/count?groupBy=folderId) isRecursive — specifies whether to count access rules recursively (e.g. api/v1/accessrules/count?isRecursive=true) isGlobal — specifies whether to count only global or only local access rules. All access rules are counted if not specified (default) (e.g. api/v1/accessrules/count?isGlobal=true)
Body	
Success Response	Status code: 200 Content-type: application/json Body: <pre>[{ "groupByValue": <String>, "count": <Number> }, ...]</pre>
Error Response	<u>400 (Invalid Request)</u> <u>401 (Unauthorized)</u> <u>403 (Access Denied)</u> <u>422 (Validation Error)</u> <u>500 (Internal Server Error)</u>

Example 1	Request: <pre>curl -X 'GET' \ 'http://192.168.0.108:9500/api/v1/accessrules/count?groupBy=name' \ -H 'Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1Y1lljoiVXNlcilslm'</pre> Response: <pre>[{ "groupByValue": "Day shift", "count": 1 }, { "groupByValue": "Night shift", "count": 1 }, { "groupByValue": "Default", "count": 1 }]</pre>
Example 2	Request: <pre>curl -X 'GET' \ 'http://192.168.0.108:9500/api/v1/accessrules/count' \ -H 'Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1Y1lljoiVXNlcilslm'</pre> Response: <pre>[{ "count": 3 }]</pre>

2.9.7. Get all folders of access rules

URL Endpoint	api/v1/accessrules/hierarchy
Method	GET
Header Parameters	Authorization: Bearer <JWT access token>
Path Parameters	
Query Parameters	name — filter by folder name with condition "starts with" (e.g. api/v1/accessrules/hierarchy?name=access) parentId — filter by parent folder ID (e.g. api/v1/accessrules/hierarchy?parentId=2) limit — specifies how many entities the response should contain (e.g. api/v1/accessrules/hierarchy?limit=10 — means that only 10 entities are returned). The value of this parameter must be between 0 and 3000 offset — specifies how many entities should be skipped from the top (e.g. api/v1/accessrules/hierarchy?offset=10 — means that 10 records are skipped, and next 10 entities are returned). The value of this parameter must be greater than or equal to 0
Body	
Success Response	Status code: 200 Content-type: application/json Body: <pre>[{ "id": <Number>, "name": <String>, "parentId": <Number>, "level": <Number> }, ...]</pre>
Error Response	400 (Invalid Request) 401 (Unauthorized) 403 (Access Denied) 500 (Internal Server Error)
Example	Request: <pre>curl -X 'GET' \ 'http://192.168.0.108:9500/api/v1/accessrules/hierarchy' \ -H 'Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilslm'</pre> Response: <pre>[{ "id": 9, "name": "Head office 5/2", "parentId": 2, "level": 2 }, { "id": 10, "name": "Head office 2/2", "parentId": 2, "level": 2 }]</pre>

2.10. Access points

2.10.1. Get all access points

URL Endpoint	api/v1/accesspoints
Method	GET
Header Parameters	Authorization: Bearer <JWT access token>
Path Parameters	
Query Parameters	▪ id - identifier(s) of Access Point(s) - returns only records with specified identifiers (e.g. api/v1/accesspoints?id=1,2 or api/v1/accesspoints?id=1&id=2) ▪ name - filter by access point name with condition "starts with" (e.g. api/v1/accesspoints?name=Door1) ▪ folderId - filter by folder id (e.g. api/v1/accesspoints?folderId=4) ▪ limit - specifies how many entities the response should contain (e.g. api/v1/accesspoints?limit=10 - means that only 10 entities are returned). The value of this parameter must be between 0 and 3000 ▪ offset - specifies how many entities should be skipped from the top (e.g. api/v1/accesspoints?limit=10&offset=10 - means that 10 records are skipped, and next 10 entities are returned). The value of this parameter must be greater than or equal to 0 ▪ includeFields - specifies what fields the response should contain (e.g. api/v1/accesspoints?includeFields=name - only field "name" is returned) ▪ excludeFields - specifies what fields the response should not contain (e.g. api/v1/accesspoints?excludeFields=id - all fields except "id" are returned) ▪ sortBy - specifies sort field (e.g. api/v1/accesspoints?sortBy=name - means that entities would be sorted by the "name" field) ▪ sortOrder - specifies sort order. Can be either ASC or DESC (e.g. api/v1/accesspoints?sortBy=name&sortOrder=DESC - means that entities would be sorted by the "name" field in descending order)
Body	
Success Response	Status code: 200 Content-type: application/json Body: <pre>[{ "id": <Number>, "name": <String>, "folderId": <Number> // access point folder ID }, ...]</pre>
Error Response	<u>400 (Invalid request)</u> <u>401 (Unauthorized)</u> <u>403 (Permission denied)</u> <u>500 (Internal Server Error)</u>

Example	Request: <pre>curl -X GET http://192.168.0.108:9500/api/v1/accesspoints -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilslm"</pre> Response: <pre>[{ "id":1, "name":"AccessPoint 1", "folderId": 1 }, { "id":2, "name":"AccessPoint 2", "folderId": 2 }]</pre>
---------	--

2.10.2. Get access point by ID

URL Endpoint	api/v1/accesspoints/{id}
Method	GET
Header Parameters	Authorization: Bearer <JWT access token>
Path Parameters	id - identifier of Access Point
Query Parameters	includeFields - specifies what fields the response should contain (e.g. api/v1/accesspoints/1?includeFields=name - only the "name" field is returned) excludeFields - specifies what fields the response should not contain (e.g. api/v1/accesspoints/1?excludeFields=id - all fields except "id" are returned)
Body	
Success Response	Status code: 200 Content-type: application/json Body: <pre>{ "id": <Number>, "name": <String>, "folderId": <Number> // access point folder ID }</pre>
Error Response	<u>400 (Invalid request)</u> <u>401 (Unauthorized)</u> <u>403 (Permission denied)</u> <u>404 (Access Point with specified id not found)</u> <u>500 (Internal Server Error)</u>
Example	Request: <pre>curl -X GET http://192.168.0.108:9500/api/v1/accesspoints/2 -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilsm"</pre> Response: <pre>{ "id": 2, "name": "AccessPoint 2", "folderId": 1 }</pre>

2.10.3. Get all folders of access points

URL Endpoint	api/v1/accesspoints/hierarchy
Method	GET
Header Parameters	Authorization: Bearer <JWT access token>
Path Parameters	
Query Parameters	▪ name - filter by name with condition "starts with" (e.g. api/v1/accesspoints/hierarchy?name=door) ▪ parentID - filter by parent id (e.g. api/v1/accesspoints/hierarchy?parentID=4) ▪ limit - specifies how many entities the response should contain (e.g. api/v1/accesspoints/hierarchy?limit=10). The value of this parameter must be between 0 and 3000 ▪ offset - specifies how many entities should be skipped from the top (e.g. api/v1/accesspoints/hierarchy?offset=10). The value of this parameter must be greater than or equal to 0
Body	
Success Response	Status code: 200 Content-type: application/json Body: <pre>[{ "id": <Number>, "name": <String>, "parentID": <Number>, // parent folder ID }, ...]</pre>
Error Response	<u>400 (Invalid request)</u> <u>401 (Unauthorized)</u> <u>403 (Access denied)</u> <u>500 (Internal Server Error)</u>
Example	Request: <pre>curl -X GET http://192.168.0.108:9500/api/v1/accesspoints/hierarchy -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilslm"</pre> Response: <pre>[{ "id": 5, "name": "1 здание", "parentID": 0, }, { "id": 2, "name": "2 здание", "parentID": 0, }]</pre>

2.10.4. Count access points

URL Endpoint	api/v1/accesspoints/count
Method	GET
Header Parameters	Authorization: Bearer <JWT access token>
Path Parameters	
Query Parameters	id - identifier(s) of Access Points(s) - returns count of records with specified identifiers (e.g. api/v1/accesspoints/count?id=1) name - filter by access rule name with condition "starts with" (e.g. api/v1/accesspoints/count?name=%D1%81) folderId - filter by folder id (e.g. api/v1/accesspoints/count?folderId=4) groupBy - specifies which field to use for grouping. Permitted values are: name, folderId (e.g. api/v1/accesspoints/count?groupBy=name) isRecursive - specifies whether to count access points recursively or not (e.g. api/v1/accesspoints/count?isRecursive=true)
Body	
Success Response	Status code: 200 Content-type: application/json Body: <pre>[{ "count": <Number> }]</pre>
Error Response	<u>400 (Invalid request)</u> <u>401 (Unauthorized)</u> <u>403 (Access denied)</u> <u>422 (Validation Error)</u> <u>500 (Internal Server Error)</u>
Example	Request: <pre>curl -X GET http://192.168.0.108:9500/api/v1//accesspoints/count -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilslm"</pre> Response: <pre>[{ "count": 3 }]</pre>

2.11. Users

2.11.1. Get all users

URL Endpoint	/api/v1/users/
Method	GET
Header Parameters	Authorization: Bearer <JWT access token>
Path Parameters	
Query Parameters	id - Identifier(s) of a user(employee) (e.g. /api/v1/users?id=5&id=6&id=7) username - Filter by username with condition "starts with" (e.g. api/v1/users?username=user) limit - Specifies how many entities the response should contain (e.g. /api/v1/users?limit=10). The value of this parameter must be between 0 and 3000 offset - Specifies how many entities should be skipped from the top (e.g. /api/v1/users?offset=2). The value of this parameter must be greater than or equal to 0 sortBy - Sort field (Default value: id) (e.g. /api/v1/users?sortBy=id) sortOrder - Sort order. Can be either ASC or DESC (Default value: ASC) (e.g. /api/v1/users?sortOrder=ASC)
Body	
Success Response	Status code: 200 Content-type: application/json Body: <pre>[{ "id": <Number>, "username": <String>, "depRestriction": <Boolean>,// indicates whether the user has restricted access to departments "depAccessList": [<Number>]// list of department IDs the user can access }, ...]</pre>
Error Response	<u>400 (Invalid request)</u> <u>401 (Unauthorized)</u> <u>403 (Permission denied)</u> <u>422 (Unprocessable Entity)</u> <u>500 (Internal Server Error)</u>
Example	Request: <pre>curl -X GET http://192.168.0.108:9500/api/v1/users -H 'accept: application/json' -H 'Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiIjojVXNlciIsIm'</pre> Response: <pre>[{ "id": 6, "username": "Administrator", "depRestriction": false, "depAccessList": [] }, { "id": 10, "username": "User", "depRestriction": false, "depAccessList": [] }]</pre>

2.11.2. Get user by ID

URL Endpoint	/api/v1/users/{id}
Method	GET
Header Parameters	Authorization: Bearer <JWT access token>
Path Parameters	id - identifier of a user (employee)
Query Parameters	
Body	
Success Response	Status code: 200 Content-type: application/json Body: <pre>[{ "id": <Number>, "username": <String>, "depRestriction": <Boolean>, // indicates whether the user has restricted access to departments "depAccessList": [<Number>] // list of department IDs the user can access }]</pre>
Error Response	400 (Invalid request) 401 (Unauthorized) 403 (Permission denied) 404 (Not Found) 500 (Internal Server Error)
Example	Request: <pre>curl -X GET http://192.168.0.108:9500/api/v1/users/7 -H 'accept: application/json' -H 'Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiIjoiVXNlcilslm'</pre> Response: <pre>{ "id": 7, "username": "User", "depRestriction": true, "depAccessList": [2] }</pre>

2.11.3. Create user from an existing employee

URL Endpoint	/api/v1/users/
Method	POST
Header Parameters	Authorization: Bearer <JWT access token>
Path Parameters	
Query Parameters	
Body	<pre>{ "employeeId": 0, "username": "string", "password": "string", "depRestriction": true, "depAccessList": [0] }</pre> You should take into account the restrictions on user passwords described in "File - Settings - Operator rights".
Success Response	Status code: 200 Content-type: application/json Body: <pre>{ "id": 0, "username": "string", "depRestriction": true, "depAccessList": [0] }</pre>
Error Response	<u>401 (Unauthorized)</u> <u>403 (Permission denied)</u> <u>422 (Unprocessable Entity)</u> <u>500 (Internal Server Error)</u>
Example	Request: <pre>curl -X POST http://127.0.0.1:9500/api/v1/users -H 'accept: application/json' -H 'Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilsm'</pre> Response: <pre>{ "id": 11, "username": "user", "depRestriction": true, "depAccessList": [2] }</pre>

2.11.4. Update user

URL Endpoint	/api/v1/users/{id}
Method	PUT
Header Parameters	Authorization: Bearer <JWT access token>
Path Parameters	id - identifier of a user (employee)
Query Parameters	
Body	<pre>{ "username": "string", "password": "string", "depRestriction": true, "depAccessList": [0] }</pre> Body contains fields that should be updated. It is possible to update the following fields: - username - password - depRestriction - depAccessList The user making this request cannot change their own settings, except for the password (the checkbox "Change your password" must be enabled in "Cardholders - Operator"). You should take into account the restrictions on user passwords described in "File - Settings - Operator rights".
Success Response	Status code: 200 Content-type: application/json Body: <pre>{ "id": 0, "username": "string", "depRestriction": true, "depAccessList": [0] }</pre>
Error Response	<u>401 (Unauthorized)</u> <u>403 (Permission denied)</u> <u>404 (Not Found)</u> <u>422 (Unprocessable Entity)</u> <u>500 (Internal Server Error)</u>
Example	Request: <pre>curl -X PUT http://192.168.0.108:9500/api/v1/users/8 -H 'accept: application/json' -H 'Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilsm'</pre> Response: <pre>{ "id": 8, "username": "Office Administrator", "depRestriction": false, "depAccessList": [] }</pre>

2.11.5. Delete user

This request revokes the right to be a system user. The employee is not deleted.

URL Endpoint	/api/v1/users/{id}
Method	DELETE
Header Parameters	Authorization: Bearer <JWT access token>
Path Parameters	id - identifier of a user (employee). The user making this request cannot revoke their own right to be a system user.
Query Parameters	
Body	
Success Response	Status code: 200 Content-type: application/json Body: <pre>{ "id": 0, "depRestriction": true, "depAccessList": [0] }</pre>
Error Response	<u>401 (Unauthorized)</u> <u>403 (Permission denied)</u> <u>404 (Not Found)</u> <u>422 (Unprocessable Entity)</u> <u>500 (Internal Server Error)</u>
Example	Request: <pre>curl -X DELETE http://192.168.0.108:9500/api/v1/users/7 -H 'accept: application/json' -H 'Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilslm'</pre> Response: <pre>{ "id": 7, "depRestriction": false, "depAccessList": [] }</pre>

2.12. Zones

2.12.1. Get all zones

URL Endpoint	api/v1/zones
Method	GET
Header Parameters	Authorization: Bearer <JWT access token>
Path Parameters	
Query Parameters	id - identifier(s) of Zone(s) - returns only records with specified identifiers (e.g. api/v1/zones?id=1,2 or api/v1/zones?id=1&id=2) accessPointId - filter by Access Point, returns only zones linked with specified Access Point (e.g. api/v1/zones?accessPointId=10) name - filter by zone name with condition "starts with" (e.g. api/v1/zones?name=ZoneA) accessPointId - identifier of access point (e.g. api/v1/zones?accessPointId=1) limit - specifies how many entities the response should contain (e.g. api/v1/zones?limit=10 - means that only 10 entities are returned). The value of this parameter must be between 0 and 3000 type - specifies what type of zones the response should contain (e.g. api/v1/zones?type=WORK) offset - specifies how many entities should be skipped from the top (e.g. api/v1/zones?limit=10&offset=10 - means that 10 records are skipped and next 10 entities are returned). The value of this parameter must be greater than or equal to 0 includeFields - specifies what fields the response should contain (e.g. api/v1/zones?includeFields=name - only the "name" field is returned) excludeFields - specifies what fields the response should not contain (e.g. api/v1/zones?excludeFields=id - all fields except "id" are returned) sortBy - specifies sort field (e.g. api/v1/zones?sortBy=name - means that entities would be sorted by the "name" field) sortOrder - specifies sort order. Can be either ASC or DESC (e.g. api/v1/zones?sortBy=type&sortOrder=ASC - means that entities would be sorted by the "type" field in ascending order (WORK -> EVACUATION -> UNDEFINED))
Body	
Success Response	Status code: 200 Content-type: application/json Body: <pre>[{ "id": <Number>, "name": <String>, "type": <Enum: [WORK, EVACUATION, UNDEFINED]>, "accessPoints": [{ "id": <Number>, "direction": <Enum: [IN, OUT]> }, ...] }, ...]</pre>
Error Response	400 (Invalid request) 401 (Unauthorized) 403 (Permission denied) 500 (Internal Server Error)

Example	Request: <pre>curl -X GET http://192.168.0.108:9500/api/v1/zones -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilslm"</pre> Response: <pre>[{ "id": 1, "name": "Zone 1", "type": "WORK", "accessPoints": [{ "id": 1, "direction": "IN" }, { "id": 2, "direction": "OUT" }] }, { "id": 2, "name": "Zone 2", "type": "EVACUATION", "accessPoints": [{ "id": 3, "direction": "IN" }, { "id": 4, "direction": "OUT" }] }]</pre>
---------	---

2.12.2. Get zone by ID

URL Endpoint	api/v1/zones/{id}
Method	GET
Header Parameters	Authorization: Bearer <JWT access token>
Path Parameters	id - identifier of zone
Query Parameters	includeFields - specifies what fields the response should contain (e.g. api/v1/zones/1?includeFields=name - only the "name" field is returned) excludeFields - specifies what fields the response should not contain (e.g. api/v1/zones/1?excludeFields=id - all fields except "id" are returned)
Body	
Success Response	Status code: 200 Content-type: application/json Body: <pre>[{ "id": <Number>, "name": <String>, "type": <Enum: [WORK, EVACUATION, UNDEFINED]>, "accessPoints": [{ "id": <Number>, "direction": <Enum: [IN, OUT]> }, ...] }]</pre>
Error Response	400 (Invalid request) 401 (Unauthorized) 403 (Permission denied) 404 (Zone with specified id not found) 422 (Validation Error) 500 (Internal Server Error)
Example	Request: <pre>curl -X GET http://192.168.0.108:9500/api/v1/zones/3 -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilslm"</pre> Response: <pre>{ "id": 3, "name": "Zone 3", "type": "WORK", "accessPoints": [{ "id": 5, "direction": "IN" }, { "id": 6, "direction": "OUT" }] }</pre>

2.13. Custom fields

2.13.1. Overview

Custom fields are additional fields that can be added to an entity and can be used for filtering.

The following entity types are currently available:

- EMPLOYEES — applies to Employees;
- VEHICLES — applies to Vehicles;
- DEPARTMENTS — applies to Departments;
- POSITIONS — applies to Positions;
- CARDS — applies to Cards;
- GUESTBINDINGS — applies to Guest Card assignments;
- GUEST_CARDS — applies to Guest Cards.

Custom fields for DEPARTMENTS, POSITIONS, and CARDS can only be created and managed via the REST API. They are not displayed in the client application.

The following data types are supported:

- STRING;
- DATE (e.g., 2021-03-21);
- DATETIME (e.g., 2021-03-21T18:00:57) (with timezone in ISO8601 format);
- TIME (e.g., 12:36);
- LONG;
- DOUBLE;
- BOOLEAN;
- LIST.



A field name can contain only English letters and numbers ([a-zA-Z0-9]).

2.13.2. Get all custom fields

URL Endpoint	api/v1/custom/fields
Method	GET
Header Parameters	Authorization: Bearer <JWT access token>
Path Parameters	
Query Parameters	▪ entity - filter by entity (e.g. api/v1/custom/fields?entity=EMPLOYEE) ▪ name - filter by custom field name with condition "starts with" (e.g. api/v1/custom/fields?name=dateBirth) ▪ limit - specifies how many entities the response should contain (e.g. api/v1/custom/fields?limit=10 - means that only 10 entities are returned). The value of this parameter must be between 0 and 3000 ▪ offset - specifies how many entities should be skipped from the top (e.g. api/v1/custom/fields?limit=10&offset=10 - means that 10 records are skipped, and next 10 entities are returned). The value of this parameter must be greater than or equal to 0 ▪ includeFields - specifies what fields the response should contain (e.g. api/v1/custom/fields?includeFields=name - only the "name" field is returned) ▪ excludeFields - specifies what fields the response should not contain (e.g. api/v1/custom/fields?excludeFields=id - all fields except "id" are returned) ▪ sortBy - specifies sort field (e.g. api/v1/custom/fields?sortBy=name - means that entities would be sorted by the "name" field) ▪ sortOrder - specifies sort order. Can be either ASC or DESC (e.g. api/v1/custom/fields?sortBy=name&sortOrder=ASC - means that entities would be sorted by the "name" field in descending order) ▪ isGlobal - specifies whether to return only global or only local custom fields. All custom fields are returned if not specified (default) (e.g. api/v1/custom/fields?isGlobal=true)
Body	
Success Response	Status code: 200 Content-type: application/json Body: <pre>[{ "id": <Number>, "name": <String>, "description": <String>, "type": <Enum [LONG, DOUBLE, STRING, DATE, DATETIME, TIME, BOOLEAN, LIST]> // data type "entity": <Enum [EMPLOYEES, VEHICLES, DEPARTMENTS, POSITIONS, CARDS, GUESTBINDINGS, GUEST_CARDS]>, // entity this field applies to "values": [<String>, <String>, ...], // possible values for LIST type "required": <Boolean>, "isGlobal": <Boolean> // indicates whether the custom field is global or local }, ...]</pre>
Error Response	400 (Invalid request) 401 (Unauthorized) 403 (Permission denied) 500 (Internal Server Error)

Example	Request: <pre>curl -X GET http://192.168.0.108:9500/api/v1/custom/fields -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilslm"</pre> Response: <pre>[{ "id":1, "name":"birthDate", "description":"Employee birth date", "type":"STRING", "entity":"EMPLOYEES", "required": false, "isGlobal": false }, { "id":2, "name":"Model", "description":"Vehicle model", "type":"STRING", "entity":"VEHICLES", "required": false, "isGlobal": false }, { "id":3, "name":"Color", "description":"Color of some attribute", "type":"LIST", "entity":"EMPLOYEES", "values":["Red", "Green", "Blue"], "required": false, "isGlobal": false }]</pre>
---------	---

2.13.3. Create custom field

URL Endpoint	api/v1/custom/fields
Method	POST
Header Parameters	Authorization: Bearer <JWT access token> Content-Type: application/json
Path Parameters	
Query Parameters	
Body	<pre>{ "name": <String>, "description": <String>, "type": <Enum (see above)>, "entity": "<Enum (see above)>", "values": [<String>, ...], "required": <Boolean> }</pre> Required fields: • name • description • type • entity • values (only if type is LIST) Attribute "required" is false by default.
Success Response	Status code: 200 Content-type: application/json Body: <pre>{ "id": <Number>, "name": <String>, "description": <String>, "type": <Enum (see above)>, "entity": "<Enum (see above)>", "values": [<String>, <String>, ...], "required": <Boolean>, "isGlobal": <Boolean> }</pre>
Error Response	401 (Unauthorized) 403 (Permission denied) 422 (Validation Error) 500 (Internal Server Error)
Example	Request <pre>curl -X POST http://192.168.0.108:9500/api/v1/custom/fields -H "Content-Type: application/json" -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ0eXB1IjoiVXNlciIsIm"-d '{"name":"birthDate","description":"Employee's date birth","type":"STRING","entity":"EMPLOYEES"}'</pre> Response: <pre>{ "id":1, "name":"birthDate", "description":"Employee birth date", "type":"STRING", "entity":"EMPLOYEES", "required": false, "isGlobal": false }</pre>

2.13.4. Update custom field

URL Endpoint	api/v1/custom/fields/{id}
Method	PUT
Header Parameters	Authorization: Bearer <JWT access token> Content-Type: application/json
Path Parameters	id - identifier of Custom Field to be edited
Query Parameters	
Body	<pre>{ "name": <String>, "description": <String>, "values": [<String>, <String>, ...], "required": <Boolean> }</pre> Body contains fields that should be modified. Only new values can be added for type LIST.
Success Response	Status code: 200 Content-type: application/json Body: <pre>{ "id": <Number>, "name": <String>, "description": <String>, "type": <Enum (see above)>, "entity": <Enum (see above)>, "values": [<String>, <String>, ...], "required": <Boolean>, "isGlobal": <Boolean> }</pre>
Error Response	401 (Unauthorized) 403 (Permission denied) 404 (Custom Field with specified id not found) 422 (Validation Error) 500 (Internal Server Error)
Example	Request: <pre>curl -X PUT http://192.168.0.108:9500/api/v1/custom/fields/1 -H "Content-Type: application/json" -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilsm" -d '{"description": "Birth Date"}'</pre> Response: <pre>{ "id": 1, "name": "birthDate", "description": "Birth date", "type": "STRING", "entity": "EMPLOYEES", "required": false, "isGlobal": false }</pre>

2.13.5. Delete custom field

URL Endpoint	api/v1/custom/fields/{id}
Method	DELETE
Header Parameters	Authorization: Bearer <JWT access token>
Path Parameters	id - identifier of Custom Field to be deleted
Query Parameters	
Body	
Success Response	Status code: 200 Content-type: application/json Body: <pre>{ "id": <Number>, "name": <String>, "description": <String>, "type": <Enum (see above)>, "entity": "Enum (see above)", "values": [<String>, <String>, ...], "required": <Boolean>, "isGlobal": <Boolean> }</pre>
Error Response	401 (Unauthorized) 403 (Permission denied) 404 (Custom Field with specified id not found) 500 (Internal Server Error)
Example	Request: <pre>curl -X DELETE http://192.168.0.108:9500/api/v1/custom/fields/1 -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilslm"</pre> Response: <pre>{ "id": 1, "name": "birthDate", "description": "Birth date", "type": "STRING", "entity": "EMPLOYEES", "required": false, "isGlobal": false }</pre>

2.14. Events

2.14.1. Get all event types

URL Endpoint	api/v1/events/types
Method	GET
Header Parameters	Authorization: Bearer <JWT access token>
Path Parameters	
Query Parameters	▪ name — filter by type name with condition "starts with" ▪ limit — specifies how many entities the response should contain (e.g. api/v1/events/types?limit=10 — means that only 10 entities are returned). The value of this parameter must be between 0 and 3000 ▪ offset — specifies how many entities should be skipped from the top (e.g. api/v1/events/types?limit=10&offset=10 — means that 10 records are skipped, and next 10 entities are returned). The value of this parameter must be greater than or equal to 0 ▪ includeFields — specifies what fields the response should contain (e.g. api/v1/events/types?includeFields=name — only the "name" field is returned) ▪ excludeFields — specifies what fields the response should not contain (e.g. api/v1/events/types?excludeFields=id — all fields except "id" are returned) ▪ sortBy — specifies sort field (e.g. api/v1/events/types?sortBy=name — means that entities would be sorted by the "name" field) ▪ sortOrder — specifies sort order. Can be either ASC or DESC (e.g. api/v1/events/types?sortBy=name&sortOrder=DESC)
Body	
Success Response	Status code: 200 Content-type: application/json Body: <pre>[{ "id":<Number>, "name":<String>, "description":<String> }, ...]</pre>
Error Response	<u>400 (Invalid request)</u> <u>401 (Unauthorized)</u> <u>403 (Permission denied)</u> <u>500 (Internal Server Error)</u>
Example	Request: <pre>curl -X GET http://192.168.0.108:9500/api/v1/events/types -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilslm"</pre> Response: <pre>[{ "id":6, "name":"passDetected2", "description":"Generated by controller when pass is registered" }, { "id":7, "name":"passDeny2", "description":"Generated by controller when access denied is registered" }]</pre>

2.14.2. Get list of events

URL Endpoint	api/v1/events
Method	GET
Header Parameters	Authorization: Bearer <JWT access token>
Path Parameters	
Query Parameters	startTime - filter by timestamp with "greater than" condition* endTime - filter by timestamp with "less than" condition* *Note that the request must include a time offset. The date and time must be separated by a "T", e.g. "2024-11-05T11:00:00+03:00". HTTP request examples: api/v1/events?startTime=2021-04-01T08%3A00%3A00Z or api/v1/events?startTime=2021-04-01T08%3A00%3A00%2B03%3A00 api/v1/events?endTime=2021-04-30T23%3A59%3A59Z or api/v1/events?endTime=2021-04-30T23%3A59%3A59%2B03%3A00 where %2B is the code for a "+", %3A is the code for a ":" eventTypeId - filter events by type - only events with specified type are returned (e.g. api/v1/events?eventTypeId=6,12) lastId - only events with identifiers greater than specified are returned (e.g. api/v1/events?lastId=122345) accessObjectId - only events for specified access object are returned (access object identifier) (e.g. api/v1/events?accessObjectId=1456) accessPointId - only events for specified Access Point are returned (e.g. api/v1/events?accessPointId =6543) limit - specifies how many entities the response should contain (e.g. api/v1/events?limit=10 - means that only 10 entities are returned). The value of this parameter must be between 0 and 3000 offset - specifies how many entities should be skipped from the top (e.g. api/v1/events?limit=10&offset=10 - means that 10 records are skipped, and next 10 entities are returned). The value of this parameter must be greater than or equal to 0 includeFields - specifies what fields the response should contain (e.g. api/v1/events?includeFields=type,timestamp - only the fields "type" and "timestamp" are returned) excludeFields - specifies what fields the response should not contain (e.g. api/v1/events?excludeFields=id - all fields except "id" are returned) sortBy - specifies sort field (e.g. api/v1/events?sortBy=timestamp - means that entities would be sorted by the "timestamp" field) sortOrder - specifies sort order. Can be either ASC or DESC (e.g. api/v1/events?sortBy=timestamp&sortOrder=DESC)
Body	
Success Response	Status code: 200 Content-type: application/json Body: <pre>[{ "id":<Number>, "type":<Number>, "timestamp":<String>, "accessPointId":<Number>, "accessObjectId":<Number>, "direction":<String> }, ...]</pre>
Error Response	400 (Invalid request) 401 (Unauthorized) 403 (Permission denied) 500 (Internal Server Error)

Example	Request: <pre>curl -X GET http://192.168.0.108:9500/api/v1/events -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilslm"</pre> Response: <pre>[{ "id":1, "type":6, "timestamp":"2021-03-31T13:46:14+03:00", "accessPointId":6543, "accessObjectId":3456, "direction":"IN" }</pre>
---------	--

2.14.3. Get all parsed events

URL Endpoint	api/v1/events/parsed
Method	GET
Header Parameters	Authorization: Bearer <JWT access token>
Path Parameters	
Query Parameters	startTime - filter by timestamp with "greater than" condition* endTime - filter by timestamp with "less than" condition* *Note that the request must include either a time offset or a time zone. If you specify a time offset, the date and time must be separated by a "T", e.g. "2024-11-05T11:00:00+03:00". If you use a time zone abbreviation, the values must be separated by spaces, e.g. "2024-11-05 11:00:00 MSK". <u>HTTP request examples:</u> api/v1/events/parsed?startTime=2024-11-05T11%3A00%3A00%2B03%3A00 or api/v1/events/parsed?startTime=2024-11-05%2011%3A00%3A00%20MSK api/v1/events/parsed?endTime=2024-11-05T11%3A00%3A00%2B03%3A00 or api/v1/events/parsed?endTime=2024-11-05%2011%3A00%3A00%20MSK where %2B is the code for a "+", %3A is the code for a ":", %20 is a space. eventType - filter by event type (e.g. api/v1/events/parsed?eventType=PASS_DETECTED) accessPointId - filter by access point id (e.g. api/v1/events/parsed?accessPointId=1) accessObjectId - filter by access object id (e.g. api/v1/events/parsed?accessObjectId=1) cardKey - filter by card key (e.g. api/v1/events/parsed?cardKey=1800000100000000) lastId - only events with identifiers greater than specified are returned (e.g. api/v1/events/parsed?lastId=12345) lastLogId - only events with identifiers greater than specified are returned (e.g. api/v1/events/parsed?lastLogId=12345) includeFields - specifies what fields the response should contain (e.g. api/v1/events/parsed?includeFields=eventType,timestamp) excludeFields - specifies what fields the response should not contain (e.g. api/v1/events/parsed?excludeFields=id) limit - specifies how many entities the response should contain (e.g. api/v1/events/parsed?limit=10). The value of this parameter must be between 0 and 3000 offset - specifies how many entities should be skipped from the top (e.g. api/v1/events/parsed?offset=10). The value of this parameter must be greater than or equal to 0
Body	
Success Response	Status code: 200 Content-type: application/json Body: <pre>{ "id": <Number>, "logId": <Number>, "data": <Object>, "additionalData": <Object>, "eventType": <Enum: [ACCESS_ABORTED, PASS_DETECTED, ...]>, "timestamp": <DateTime>, "receivedTime": <DateTime> } ...]</pre>
Error Response	400 (Invalid Request) 401 (Unauthorized) 403 (Access Denied) 500 (Internal Server Error)

Example	Request: <pre>curl -X GET http://localhost:9500/api/v1/events/parsed -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilslm"</pre> Response: <pre>[{ "id": 1, "logId": 100, "data": { "accessPointId": 1, "direction": "IN", "passReasonCode": 5, "passReason": "Pass through open door", "employeeId": 45, "cardKey": "1800000100000000", "alcValue": 0.1, "zoneId": 1 }, "additionalData": { "accessObject": { "type": "EMPLOYEE", "data": { "id": 45, "name": "John Doe", "position": "Software Developer" } }, "zone": { "id": 1, "name": "Zone A" }, "accessPoint": { "id": 1, "name": "AP1", "linkType": "ETHACTIVE" } }, "eventType": "PASS_DETECTED", "timestamp": "2021-11-30T18:00:00+03:00", "receivedTime": "2021-11-30T18:00:01+03:00" }]</pre>
---------	---

Possible content of "data"

Type of event or group of events	Description	Content of "data"
ACCESS_ABORTED	Pass aborted due to timeout	Success response <pre>... "data": { "zoneld": <NUMBER>, "accessPointId": <NUMBER>, "direction": <STRING>, "employeeId": <NUMBER>, "cardKey": <STRING> } ...</pre> Example <pre>... "data": { "zoneld": 1, "accessPointId": 1, "direction": "IN", "employeeId": 5, "cardKey": "20E1282CC6000000" } ...</pre>
PASS_DETECTED	Pass registered	Success response <pre>... "data":{ "zoneld": <NUMBER>, "accessPointId": <NUMBER>, "direction": <STRING>, "passReasonCode": <NUMBER>, "passReason": <STRING>, "employeeId": <NUMBER>, "cardKey": <STRING>, "alcValue": <NUMBER> } ...</pre> Example <pre>... "data":{ "zoneld": 1, "accessPointId": 1, "direction": "IN", "passReasonCode": 3, "passReason": "Pass (by rules)", "employeeId": 5, "cardKey": "20E1282CC6000000", "alcValue": 0 } ...</pre> Format of the "alcValue" field: per mille.

Type of event or group of events	Description	Content of "data"
PASS_DENY	Access denied	Success response <pre>... "data":{ "zoneld": <NUMBER>, "accessPointId": <NUMBER>, "direction": <STRING>, "denyReasonCode": <NUMBER>, "denyReason": <STRING>, "employeeId": <NUMBER>, "cardKey": <STRING>, "alcValue": <NUMBER> } ...</pre> Example <pre>... "data":{ "zoneld": 1, "accessPointId": 1, "direction": "IN", "denyReasonCode": 3, "denyReason": "Unknown key", "employeeId": 5, "cardKey": "20E1282CC6000000", "alcValue": 0 } ...</pre> Format of the "alcValue" field: per mille.
FIRE_UNLOCK_BEGIN	Fire alarm! Emergency unlock activated	Success response <pre>... "data":{ "accessPointId": <NUMBER> } ...</pre>
FIRE_UNLOCK_END	Fire alarm ended	Example <pre>... "data":{ "accessPointId": 1 } ...</pre>
DOOR_HOLD_BEGIN	Door hold	<pre>... "data":{ "accessPointId": 1 } ...</pre>
DOOR_HOLD_END	Door hold ended	<pre>...</pre>
BOX_CLOSED	Controller case closed	Success response <pre>... "data":{ "accessPointId": <NUMBER>, "numberOfAccessPointsOnTheController": <NUMBER> } ...</pre> Example <pre>... "data":{ "accessPointId": 4, "numberOfAccessPointsOnTheController": 1 } ...</pre>

Type of event or group of events	Description	Content of "data"
BOX_OPENED	Controller case opened	Success response <pre>... "data":{ "accessPointId": <NUMBER>, "numberOfAccessPointsOnTheController": <NUMBER> } ...</pre> Example <pre>... "data":{ "accessPointId": 4, "numberOfAccessPointsOnTheController": 1 } ...</pre>
AP_ONLINE_STATUS	Connection to the access point lost or restored	Success response <pre>... "data":{ "accessPointId": <NUMBER>, "onlineStatus": <NUMBER> } ...</pre> Example <pre>... "data":{ "accessPointId": 4, "onlineStatus": 1 } ...</pre>
DEV_ACTION	Escort card read or gate state change	Success response <pre>... "data":{ "accessPointId": <NUMBER>, "direction": <STRING>, "action": <STRING>, "zoneId": <NUMBER>, "employeeId": <NUMBER>, "cardKey": <STRING> } ...</pre> Example <pre>... "data":{ "accessPointId": 1, "direction": "IN", "action": "Escort card read", "zoneId": 1, "employeeId": 5, "cardKey": "20E1282CC6000000" } ...</pre>

Type of event or group of events	Description	Content of "data"
VOLTAGE_STATUS This event type has been deprecated as of Sigur software version 1.6.8.101. See the DC_STATUS and BATTERY_STATUS event types below.	Controller power status change	Success response <pre>... "data": { "accessPointId": <NUMBER>, "numberOfAccessPointsOnTheController": <NUMBER>, "voltageStatus": <STRING>, "batteryStatus": <STRING> } ...</pre> The "batteryStatus" field is returned only by controllers with recent firmware. Example <pre>... "data": { "accessPointId": 4, "numberOfAccessPointsOnTheController": 1, "voltageStatus": "BATTERY", "batteryStatus": "FAILURE" } ...</pre>
VOLTAGE_VALUE	Change in controller supply voltage	Success response <pre>... "data": { "accessPointId": <NUMBER>, "status": <STRING>, "value": <NUMBER>, "numberOfAccessPointsOnTheController": <NUMBER> } ...</pre> Example <pre>... "data": { "accessPointId": 1, "status": "BELOW NORMAL", "value": 115, "numberOfAccessPointsOnTheController": 1 } ...</pre> Format of the "value" field: volts.
DC_STATUS	Controller power status change	Success response <pre>... "data": { "accessPointId": <NUMBER>, "numberOfAccessPointsOnTheController": <NUMBER>, "dcStatus": <STRING> } ...</pre> Example <pre>... "data": { "accessPointId": 1, "numberOfAccessPointsOnTheController": 1, "dcStatus": "BATTERY" } ...</pre>

Type of event or group of events	Description	Content of "data"
BATTERY_STATUS	Battery status change	Success response <pre> ... "data": { "accessPointId": <NUMBER>, "numberOfAccessPointsOnTheController": <NUMBER>, "batteryStatus": <STRING> } ... </pre> Example <pre> ... "data": { "accessPointId": 1, "numberOfAccessPointsOnTheController": 1, "batteryStatus": "FAILURE" } ... </pre>
LPR_NUMBER_EVENT	License plate number detected	Success response <pre> ... "data": { "zoneId": <NUMBER>, "direction": <STRING>, "accessPointId": <NUMBER>, "verticalDirection": <STRING>, "licencePlateNumber": <STRING> } ... </pre> Example <pre> ... "data": { "zoneId": 0, "direction": "IN", "accessPointId": 4, "verticalDirection": "NO DATA", "licencePlateNumber": "A123AA12" } ... </pre>
MNG_STATE_CHANGED	Access point mode change	Success response <pre> ... "data": { "accessPointId": <NUMBER>, "oldMode": <STRING>, "newMode": <STRING>, "changeReason": <STRING>, "cardKey": <STRING> } ... </pre> Example <pre> ... "data": { "accessPointId": 1, "oldMode": "NORMAL", "newMode": "LOCKED", "changeReason": "SERVER COMMAND", "cardKey": "0" } ... </pre>

Type of event or group of events	Description	Content of "data"
AP_MODE_DENY	Failed to change access point mode	Success response <pre>... "data": { "accessPointId": <NUMBER>, "employeeId": <NUMBER>, "failReasonCode": <NUMBER>, "failReason": <STRING> } ...</pre> Example <pre>... "data": { "accessPointId": 1, "employeeId": 0, "failReasonCode": 2, "failReason": "INCORRECT_PIN" } ...</pre>
TEXT	Text message	Success response <pre>... "data":{ "accessPointId": <NUMBER>, "text": <STRING> } ...</pre> Example <pre>... "data":{ "accessPointId": 1, "text": "message" } ...</pre>
TEXT2	Alarm text message	Success response <pre>... "data":{ "accessPointId": <NUMBER>, "alarmingEvent": <BOOLEAN>, "text": <STRING> } ...</pre> Example <pre>... "data":{ "accessPointId": 1, "alarmingEvent": true, "text": "message" } ...</pre>

Type of event or group of events	Description	Content of "data"
ALKO_DINGO_RAW_REPORT	Dingo breathalyzer raw report	Success response <pre>... "data":{ "accessPointId": <NUMBER>, "direction": <STRING>, "reportData": <NUMBER> } ...</pre> Example <pre>... "data":{ "accessPointId": 1, "direction": "IN", "reportData": 6 } ...</pre>
WAITING_FOR_RULE_STAGE	Waiting for escort, PIN, alko events, confirmation of zone arming or another event	Success response <pre>... "data": { "zoneId": <NUMBER>, "cardKey": <STRING>, "direction": <STRING>, "employeeId": <NUMBER>, "accessPointId": <NUMBER>, "waitingReason": <STRING>, "alarmZoneId": <NUMBER> } ...</pre> The "alarmZoneId" parameter is returned only when "waitingReason" is "Waiting for confirmation of zone arming". Example <pre>... "data": { "zoneId": 1, "cardKey": "208F1C0855000000", "direction": "IN", "employeeId": 5, "accessPointId": 4, "waitingReason": "Waiting for escort" } ...</pre>
LOCK_FAIL	Hall sensor alarm event	Success response <pre>... "data": { "alarmType": <STRING>, "accessPointId": <NUMBER> } ...</pre> Example <pre>... "data": { "alarmType": "Hall sensor in active (should be inactive since lock is inactive)", "accessPointId": 4 } ...</pre>

Type of event or group of events	Description	Content of "data"
TEMPERATURE_FINE	Normal temperature	Success response ... "data": { "zonelId": <NUMBER>, "direction": <STRING>, "employeeId": <NUMBER>, "accessPointId": <NUMBER>, "temperature": <NUMBER>, "temperatureThreshold": <NUMBER> } ... Example ... "data": { "zonelId": 1, "direction": "IN", "employeeId": 5, "accessPointId": <NUMBER>, "temperature": 36.6, "temperatureThreshold": 37.0 } ... Format of the "temperature" and "temperatureThreshold" fields: degrees Celsius.
TEMPERATURE_WARNING	Temperature warning threshold exceeded	
TEMPERATURE_ALERT	Temperature alert threshold exceeded	
FACE_RECOGNIZED	Face identified (in identification mode)	Success response ... "data": { "zonelId": <NUMBER>, "direction": <STRING>, "employeeId": <NUMBER>, "accessPointId": <NUMBER> } ... Example ... "data": { "zonelId": 0, "direction": "IN", "employeeId": 6, "accessPointId": 4 } ...
FACE_VERIFICATION_FAILED	Face not recognized (in soft verification mode)	Success response ... "data": { "zonelId": <NUMBER>, "direction": <STRING>, "employeeId": <NUMBER>, "accessPointId": <NUMBER> } ... Example ... "data": { "zonelId": 0, "direction": "IN", "employeeId": 6, "accessPointId": 4 } ...

Type of event or group of events	Description	Content of "data"
FACE_MASK_VERIFICATION_FAILED	Face mask is absent (in soft verification mode)	Success response <pre>... "data": { "zoneId": <NUMBER>, "direction": <STRING>, "employeeId": <NUMBER>, "accessPointId": <NUMBER> } ...</pre> Example <pre>... "data": { "zoneId": 1, "direction": "IN", "employeeId": 5, "accessPointId": 1 } ...</pre>
FACE_MASK_VERIFICATION_SUCCESS	Face mask is on	Success response <pre>... "data": { "zoneId": <NUMBER>, "direction": <STRING>, "employeeId": <NUMBER>, "accessPointId": <NUMBER> } ...</pre> Example <pre>... "data": { "zoneId": 1, "direction": "IN", "employeeId": 5, "accessPointId": 1 } ...</pre>
TEMPERATURE_VERIFICATION_FAILED	Temperature check failed (in soft verification mode)	Success response <pre>... "data": { "zoneId": <NUMBER>, "direction": <STRING>, "employeeId": <NUMBER>, "accessPointId": <NUMBER> } ...</pre> Example <pre>... "data": { "zoneId": 1, "direction": "IN", "employeeId": 5, "accessPointId": 1 } ...</pre>

Type of event or group of events	Description	Content of "data"
ALARM_LINE_DENY	Failed to arm alarm zone	Success response <pre>... "data": { "accessPointId": <NUMBER>, "employeeId": <NUMBER>, "alarmZoneId": <NUMBER>, "failReasonCode": <NUMBER>, "failReason": <STRING> } ...</pre> Example <pre>... "data": { "accessPointId": 8, "employeeId": 58, "alarmZoneId": 8, "failReasonCode": 16, "failReason": "INCORRECT_PIN" } ...</pre>
ALARM_LINE_STATUS	Alarm zone status change	Success response <pre>... "data": { "accessPointId": <NUMBER>, "employeeId": <NUMBER>, "alarmZoneId": <NUMBER>, "oldState": <STRING>, "newState": <STRING>, "armingMethod": <STRING> } ...</pre> The "armingMethod" parameter is returned only when an alarm zone is armed via the Sigur MK100 reader, and "newState" has the value "ARMED". Example <pre>... "data": { "accessPointId": 4, "employeeId": 7, "alarmZoneId": 15, "oldState": "ARMING_DELAY", "newState": "ARMED", "armingMethod": "PIN" } ...</pre>

Type of event or group of events	Description	Content of "data"
BIOSMART_PALMJET_EVENTS	BioSmart PalmJet connection status	Success response <pre>... "data": { "zoneld": <NUMBER>, "accessPointId": <NUMBER>, "deviceSN": <STRING>, "direction": <STRING>, "event": <STRING> } ...</pre> Example <pre>... "data": { "zoneld": 0, "accessPointId": 1006, "deviceSN": "650250", "direction": "IN", "event": "DEVICE_ONLINE" } ...</pre>
READER_POWER_STATE	Reader power status change	Success response <pre>... "data": { "accessPointId": <NUMBER>, "direction": <STRING>, "port": <NUMBER>, "powerState": <STRING>, "power": <NUMBER> } ...</pre> Example <pre>... "data": { "accessPointId": 35, "direction": "IN", "port": 1, "powerState": "UNDER_MIN", "power": 11.8 } ...</pre> Format of the "power" field: volts.
READER_TAMPER_STATE	Reader tamper sensor state change	Success response <pre>... "data": { "accessPointId": <NUMBER>, "direction": <STRING>, "port": <NUMBER>, "tamperState": <STRING> } ...</pre> Example <pre>... "data": { "accessPointId": 35, "direction": "IN", "port": 1, "tamperState": "DETACHED" } ...</pre>

Type of event or group of events	Description	Content of "data"
READER_TEMPERATURE_STATE	Reader case temperature change	Success response <pre>... "data": { "accessPointId": <NUMBER>, "direction": <STRING>, "port": <NUMBER>, "temperatureState": <STRING>, "temperature": <NUMBER> } ...</pre> Example <pre>... "data": { "accessPointId": 35, "direction": "IN", "port": 1, "temperatureState": "OVER_MAX", "temperature": 41.5 } ...</pre> Format of the "temperature" field: degrees Celsius.
CONTROLLER_BOOTUP	Controller restart	Success response <pre>... "data": { "accessPointId": <NUMBER>, "reasonCode": <NUMBER>, "reason": <STRING> } ...</pre> Example <pre>... "data": { "accessPointId": 1, "reasonCode": 4, "reason": "Reset initiated by the controller itself" } ...</pre>

Field values

The values below are returned exactly as shown, including known misspellings.

Possible values of the "direction" field:

- IN;
- OUT;
- NOT DEFINED;
- UNKNOWN.

Possible values of the "passReasonCode" and "passReason" fields:

- 0 = Break-in;
- 1 = Pass in unlocked mode;
- 2 = Pass allowed by button;
- 3 = Pass (by rules);
- 4 = Pass (authorized by an external entity);
- 5 = Pass through open door;
- 6 = Pass (registered at work time terminal without access check);
- 7 = Pass, card was used (anviz);
- 8 = Pass, password was used (anviz);
- 9 = Pass, card + password were used (anviz);
- 128 = Waybill pass;
- Unknown reason.

Possible values of the "denyReasonCode" and "denyReason" fields:

- 0 = Wrong PIN;
- 1 = Key has expired;
- 2 = System error (code2);
- 3 = Unknown key;
- 4 = According to time zones;
- 5 = No access to this door;
- 6 = No access at this time;
- 7 = Passback;
- 8 = Main reader is used (while extra is expected);
- 9 = Extra reader is used (while main is expected);
- 10 = Time zones intersection error;
- 11 = Door is locked;
- 12 = Block button is active;
- 13 = Another airlock door is open;
- 14 = Zone capacity limit reached;
- 15 = Guard has denied the access;
- 16 = Alcohol overdose;
- 17 = Authorization timed out;
- 18 = Breathalyzer timed out;
- 19 = Escort card timed out;
- 20 = Guard decision timed out;
- 21 = Second card timed out;
- 22 = Previous transaction is in progress (code 22);
- 23 = Previous transaction is in progress;
- 24 = Unable to withdraw item cost;
- 25 = No licence plate recognition event;
- 26 = Special restrictions apply;
- 27 = There are assigned items;
- 28 = Face is not identified;
- 29 = Key brute force;
- 30 = Waiting for the escort;
- 31 = Access by LP is denied by the time zone;
- 32 = Got no decision from the external system;
- 33 = Forbidden Bby the external system;
- 34 = No connection with the server;
- 35 = Temperature check failed;
- 36 = Verification failed;
- 37 = Face mask is absent;
- 38 = Access period expired or not started;
- 39 = Waiting for confirmation of access point mode change;
- 40 = Waiting for PIN timed out;
- 41 = Same access object pass;
- 42 = Escort access object was denied;
- Unknown reason.

Possible values of the "action" field:

- Escort cart read;
- Gate closed;
- Gate opened;
- Gate moved;
- unknown action. code N.

Possible values of the "voltageStatus" field:

- MAIN SUPPLY;
- BATTERY.

Possible values of the "dcStatus" field:

- MAIN SUPPLY;
- BATTERY.

Possible values of the "batteryStatus" field:

- OK;
- FAILURE.

Possible values of the "verticalDirection" field:

- NO DATA;
- DOWN;
- UP;
- UNKNOWN.

Possible values of the "newMode" and "oldMode" fields:

- NORMAL;
- LOCKED;
- UNLOCKED;
- UNKNOWN.

Possible values of the "changeReason" field:

- UNKNOWN;
- FIRE UNLOCK;
- BUTTON;
- DOUBLE CARD READ;
- SERVER COMMAND;
- ON SCHEDULE;
- PIN;
- PIN AND PASS.

Possible values of the "failReasonCode" and "failReason" (AP_MODE_DENY) fields:

- 1 = READER_BUSY;
- 2 = INCORRECT_PIN;
- 3 = PASS_DENY;
- 4 = CONFIRMATION_TIMEOUT;
- 16 = ACCESS_POINT_NORMAL_MODE;
- 17 = ACCESS_POINT_LOCKED;
- 18 = ACCESS_POINT_UNLOCKED;
- UNKNOWN.

Possible values of the "waitingReason" field:

- Waiting for escort;
- Waiting for PIN;
- Waiting for breathalyzer;
- Waiting for operator (access is allowed);
- Waiting for operator (access id denied);
- Waiting for second access object;
- Waiting for confirmation of zone arming;
- Waiting for confirmation of access point mode change;
- Checking for face mask;
- Waiting for temperature measurement;
- Waiting for face;
- Unknown reason.

Possible values of the "alarmType" field:

- Hall sensor is inactive (should be active since lock is active, door closed);
- Hall sensor is active (should be inactive since lock is inactive);
- Unknown.

Possible values of the "status" field:

- NORMAL;
- BELOW NORMAL;
- ABOVE NORMAL;
- UNKNOWN.

Possible values of the "onlineStatus" field:

- 1 (connection success);
- 0 (connection failed).

Possible values of the "failReasonCode" and "failReason" (ALARM_LINE_DENY) fields:

- 1 = ALARM_LINE_NOT_CONFIGURED;
- 2 = READER_BUSY;
- 16 = INCORRECT_PIN;
- 17 = CONFIRMATION_TIMEOUT;
- 18 = PASS_DENY;
- 32 = ALARM;
- 33 = ALREADY_ARMING;
- 34 = ALARMLINE_RESISTANCE_ABNORMAL;
- 35 = DOOR_OPEN;
- 36 = GATES_OPEN;
- UNKNOWN.

Possible values of the "armingMethod" field:

- PASS_HOLDING;
- PIN;
- PIN_AND_PASS;
- UNKNOWN.

Possible values of the "oldState", "newState" fields:

- NOT_CONFIGURED;
- DISARMED;
- ARMING_DELAY (alarm zone is being armed);
- ARMED;
- FIRED;
- UNKNOWN.

Possible values of the "event" field:

- DEVICE_ONLINE;
- DEVICE_OFFLINE.

Possible values of the "powerState" and "temperatureState" fields:

- GOOD;
- UNDER_MIN;
- OVER_MAX;
- BAD;
- UNKNOWN.

Possible values of the "tamperState" field:

- INSTALLED;
- DETACHED;
- UNKNOWN.

Possible values of the "reasonCode" and "reason" fields:

- 0 = Unknown;
- 1 = Power-on reset;
- 3 = Brownout reset;
- 4 = Reset initiated by the controller itself;

- 5 = CPU reset;
- 6 = Window watchdog (WWDG1) reset;
- 7 = Independent watchdog (IWDG1) reset;
- 8 = D1 exited DStandby mode;
- 10 = D1 exited DStandby mode;
- 11 = D2 exited DStandby mode;
- 12 = Reset due to an attempt to enter CStop or D1 DStandby mode;
- 13 = Reset due to an attempt to enter D2 DStandby mode;
- Unknown reason.

2.15. Guests

2.15.1. Get guests

URL Endpoint	api/v1/guests
Method	GET
Header Parameters	Authorization: Bearer <JWT access token>
Path Parameters	
Query Parameters	• id – identifier(s) of Guest(s). Returns only records with the specified identifiers (e.g. api/v1/guests?id=1,2 or api/v1/guests?id=1&id=2) • guestCardId – identifier(s) of Guest Card(s). Returns only records with the specified identifiers (e.g. api/v1/guests?guestCardId=1,2 or api/v1/guests?guestCardId=1&guestCardId=2) • name – filter by Guest's name with condition "starts with" (e.g. api/v1/guests?name=Jack) • lpNumber – filter by license plate numbers with condition "starts with" (e.g. api/v1/guests?lpNumber=Jack) • applicationId – Guest badge application ID (0 if no application was created) (e.g. api/v1/guests?applicationId=14) • startTime[operation] – filter by Guest Card start date and time. Possible values for [operation]: EQ, GT, LT, GTE, LTE. (e.g. api/v1/guests?startTime%5Bgte%5D=2023-05-10T00:00:00Z) • expirationTime[operation] – filter by Guest Card end date and time. Possible values for [operation]: EQ, GT, LT, GTE, LTE. (e.g. api/v1/guests?expirationTime%5Bgte%5D=2023-05-10T00:00:00Z) • Both ISO 8601 and YYYY-MM-DD hh:mm:ss date-time formats are accepted in requests, but the response is always returned in ISO 8601 format. • includePhotoPreview – indicates whether to return a Guest's photo preview in the response or not. Possible values: true, false (e.g. api/v1/guests?includePhotoPreview=true) • limit – specifies how many entities the response should contain (e.g. api/v1/guests?limit=10 – means that only 10 entities are returned). The value of this parameter must be between 0 and 3000 • offset – specifies how many entities should be skipped from the top (e.g. api/v1/guests?limit=10&offset=10 – means that 10 records are skipped, and next 10 entities are returned). The value of this parameter must be greater than or equal to 0 • sortBy – specifies sort field (e.g. api/v1/guests?sortBy=name – means that entities would be sorted by "name" field) • sortOrder – specifies sort order. Can be either ASC or DESC (e.g. api/v1/guests?sortBy=name&sortOrder=DESC – means that entities would be sorted by "name" field in descending order) • includeFields – specifies what fields the response should contain (e.g. api/v1/guests?includeFields=name,guestCardId – only fields "name" and "guestCardId" are returned) • excludeFields – specifies what fields should be omitted from the response (e.g. api/v1/guests?excludeFields=description – all fields except "description" are returned)
Body	

Success Response	Status code: 200 Content-type: application/json Body: <pre>[{ "id": <Number>, "type": <Enum [QR, GUESTCARD]>, "guestCardId": <Number>, "guestInfo": { "name": <String>, "birthDate": <DateTime>, "address": <String>, "lpNumber": <String>, // guest license plate number "docType": <String>, // guest document type "docSeries": <String>, // guest document series "docNumber": <String>, // guest document number "hostEmployeeId": <Number>, // ID of the employee being visited "whoIssued": <String>, // guest document issuing authority (issued by) "email": <String>, "issuedDate": <Date>, // guest document issue date (issued on) "photoPreview": <String> // guest photo (Base64-encoded) }, "startTime": <DateTime>, // guest badge (card or QR code) validity start "expirationTime": <DateTime>, // guest badge (card or QR code) validity end "description": <String>, "applicationId": <Number> // number of guest badge application }]</pre>
Error Response	<u>400 (Invalid Request)</u> <u>401 (Unauthorized)</u> <u>403 (Permission Denied)</u> <u>422 (Validation Error)</u> <u>500 (Internal Server Error)</u>
Example	Request: <pre>curl -X GET http://192.168.0.108:9500/api/v1/guests -H "Content-Type: application/json" -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilslm"</pre> Response: <pre>[{ "id": 2, "type": "GUESTCARD", "guestCardId": 7, "guestInfo": { "name": "Jonh Doe", "birthDate": "1970-01-01", "address": "24 Buyuk Ipak Yuli Street, Tashkent, Uzbekistan", "lpNumber": "", "docType": "PASSPORT_UZ", "docSeries": "FA", "docNumber": "3254791", "hostEmployeeId": 6, "whoIssued": "Tashkent City Department of Internal Affairs", "email": "example@example.com", "issuedDate": "2020-01-01" }, "startTime": "2025-10-08T17:38:00", "expirationTime": "2025-10-09T01:38:00", "description": "Main office full access", "applicationId": 0 }]</pre>

2.15.2. Create guests with cards

You can use this endpoint only if the option "Do not permit to issue guest badges if the application No. is missing" is **disabled** in the "Client" tool (under "File – Settings – Guest badges").

Issuing a Guest Card automatically grants access to all access points. Guests' access rights must be managed through [access rules](#).

URL Endpoint	api/v1/guests/guestcards/assign
Method	POST
Header Parameters	Authorization: Bearer <JWT access token> Content-Type: application/json
Path Parameters	
Query Parameters	replace — defines how to handle a Guest Card if it is already assigned. Possible values: - true — reassigns the Guest Card to another Guest, closing the previous entry with the current expiration date. - false (default) — returns an error message (HTTP 422). (e.g. api/v1/guests/guestcards/assign?replace=true)
Body	<pre>[{ "guestCardId": <Number>, "guestInfo": { "name": <String>, "birthDate": <Date>, "address": <String>, "lpNumber": <String>, "docType": <String>, "docSeries": <String>, "docNumber": <String>, "hostEmployeeId": <Number>, "wholIssued": <String>, "email": <String>, "issuedDate": <Date>, "photo": <String> }, "startTime": <DateTime>, "expirationTime": <DateTime>, "description": <String> }, ...]</pre> Required fields: - guestCardId - name - docType — only active document types can be used (i.e., those with the "Show in document types" option enabled in the "Client" tool under "File – Settings – Documents") Optional fields: - startTime — if not set, the current date and time are used - expirationTime — if not set, it is calculated as startTime plus the "Default guest badge expiration time" value configured in the "Client" tool (under "File – Settings – Guest badges") Both ISO 8601 and YYYY-MM-DD hh:mm:ss date-time formats are accepted in requests, but the response is always returned in ISO 8601 format.

	Status code: 200 Content-type: application/json Body: <pre>[{ "id": <Number>, "type": <String>, "guestCardId": <Number>, "guestInfo": { "name": <String>, "birthDate": <Date>, "address": <String>, "lpNumber": <String>, "docType": <String>, "docSeries": <String>, "docNumber": <String>, "hostEmployeeId": <Number>, "wholIssued": <String>, "email": <String>, "issuedDate": <Date>, "photo": <String> }, "startTime": <DateTime>, "expirationTime": <DateTime>, "description": <String>, "applicationId": <Number> }, ...]</pre>
Error Response	400 (Invalid Request) 401 (Unauthorized) 403 (Permission Denied) 422 (Validation Error) 500 (Internal Server Error)
Example	Request: <pre>curl -X POST http://192.168.0.108:9500/api/v1/guests/guestcards/assign -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlciIsIm" -d '{"guestCardId":7,"guestInfo":{"name": "Jack Dawson", "docType": "PASSPORT_BY"}}'</pre> Response: <pre>{ "id": 10, "type": "GUESTCARD", "guestCardId": 7, "guestInfo": { "name": "Jack Dawson", "docType": "PASSPORT_BY" }, "startTime": "2025-10-09T09:32:59.979618", "expirationTime": "2025-10-09T18:32:59" }</pre>

2.15.3. Update guests

URL Endpoint	api/v1/guests/update
Method	POST
Header Parameters	Authorization: Bearer <JWT access token> Content-Type: application/json
Path Parameters	
Query Parameters	
Body	<pre>[{ "id": <Number>, "guestCardId": <Number>, "guestInfo": { "name": <String>, "birthDate": <Date>, "address": <String>, "lpNumber": <String>, "docType": <String>, "docSeries": <String>, "docNumber": <String>, "hostEmployeeId": <Number>, "whoIssued": <String>, "email": <String>, "issuedDate": <Date>, "photo": <String> }, "startTime": <DateTime>, "expirationTime": <DateTime>, "description": <String> }, ...]</pre> Body contains fields that should be modified. Both ISO 8601 and YYYY-MM-DD hh:mm:ss date-time formats are accepted in requests, but the response is always returned in ISO 8601 format. If any of the attributes "docSeries", "docNumber", "issuedDate", or "whoIssued" are provided, the "docType" attribute must also be specified. Only active document types can be used (i.e., those with the "Show in document types" option enabled in the "Client" tool under "File — Settings — Documents").

Success Response	Status code: 200 Content-type: application/json Body: <pre>[{ "id": <Number>, "type": <String>, "guestCardId": <Number>, "guestInfo": { "name": <String>, "birthDate": <Date>, "address": <String>, "lpNumber": <String>, "docType": <String>, "docSeries": <String>, "docNumber": <String>, "hostEmployeeId": <Number>, "wholssued": <String>, "email": <String>, "issuedDate": <Date>, "photo": <String> }, "startTime": <DateTime>, "expirationTime": <DateTime>, "description": <String>, "applicationId": <Number> }, ...]</pre>
Error Response	<u>400 (Invalid Request)</u> <u>401 (Unauthorized)</u> <u>403 (Permission Denied)</u> <u>422 (Validation Error)</u> <u>500 (Internal Server Error)</u>
Example	Request: <pre>curl -X POST http://192.168.0.108:9500/api/v1/guests/update -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilsm" -d {"id": 11, "startTime": "2025-10-09T06:35:07", "expirationTime": "2025-10-09T14:35:07"}</pre> Response: <pre>{ "id": 11, "type": "GUESTCARD", "guestCardId": 7, "guestInfo": { "name": "Jack Dawson", "docType": "PASSPORT_BY" }, "startTime": "2025-10-09T06:35:07", "expirationTime": "2025-10-09T14:35:07" }</pre>

2.15.4. Remove guests

Withdraws Guest Cards or QR Codes from Guests. The Guest Cards are not deleted.

URL Endpoint	api/v1/guests/delete
Method	POST
Header Parameters	Authorization: Bearer <JWT access token> Content-Type: application/json
Path Parameters	
Query Parameters	
Body	<pre>[<Number>, ...]</pre> In the request body, provide a list of IDs of previously created Guests.
Success Response	Status code: 204 No Content
Error Response	400 (Invalid Request) 401 (Unauthorized) 403 (Permission Denied) 422 (Validation Error) 500 (Internal Server Error)
Example	Request: <pre>curl -X POST http://192.168.0.108:9500/api/v1/guests/delete -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilslm" -d '[11, 12]'</pre> Response: 204 No Content

2.15.5. Get guest document types

URL Endpoint	api/v1/guests/doctypes
Method	GET
Header Parameters	Authorization: Bearer <JWT access token>
Path Parameters	
Query Parameters	• includeInactive — defines whether inactive document types should also be returned (i.e., those with the "Show in document types" option disabled in the "Client" tool under "File — Settings — Documents"). Possible values: true, false (default) (e.g. api/v1/guests/doctypes?includeInactive=true)
Body	
Success Response	Status code: 200 Content-type: application/json Body: <pre>[{ "docType": <String>, "isDefault": <Boolean>, // indicates whether the document type is predefined by the system "isActive": <Boolean> // indicates whether the document type is available when issuing a guest badge }]</pre>
Error Response	400 (Invalid Request) 401 (Unauthorized) 403 (Permission Denied) 500 (Internal Server Error)
Example	Request: <pre>curl -X GET http://192.168.0.108:9500/api/v1/guests/doctypes -H "Content-Type: application/json" -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlciIsIm"</pre> Response: <pre>[{ "docType": "PASSPORT_RU", "isDefault": true, "isActive": true }, { "docType": "DRIVERLICENSE_RU", "isDefault": true, "isActive": true }, { "docType": "FOREIGN_PASSPORT_RU", "isDefault": true, "isActive": true }]</pre>

2.15.6. Get guest full photo by ID

URL Endpoint	api/v1/guests/{id}/photo
Method	GET
Header Parameters	Authorization: Bearer <JWT access token>
Path Parameters	id — ID of a Guest
Query Parameters	
Body	
Success Response	Status code: 200 Content-type: image/jpeg Body: image/jpeg The response body contains the raw image bytes.
Error Response	400 (Invalid Request) 401 (Unauthorized) 403 (Permission Denied) 404 (Guest Photo for the Specified Guest ID Not Found) 500 (Internal Server Error)
Example	Request: <pre>curl -X GET http://192.168.0.108:9500/api/v1/guests/13/photo -H "Content-Type: application/json" -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJyY1ljojVXNlcilslm"</pre> Response: image/jpeg

2.16. Guest cards

2.16.1. Get all guest cards

URL Endpoint	api/v1/guestcards
Method	GET
Header Parameters	Authorization: Bearer <JWT access token>
Path Parameters	
Query Parameters	• id — identifiers of Guest Cards. Returns only records with the specified identifiers (e.g. api/v1/guestcards?id=1,2 or api/v1/guestcards?id=1&id=2) • name — filter by Guest Card's name with condition "starts with" (e.g. api/v1/guestcards?name=Badge1) • value — filter by Guest Card's value (HEX) with condition "starts with" (e.g. api/v1/guestcards?value=E6D5F6) • startTime[operation] — filter by Guest Card's start date and time. Possible values for [operation]: EQ, GT, LT, GTE, LTE. (e.g. api/v1/guestcards?startTime%5Bgte%5D=2023-05-10T00:00:00Z) • expirationTime[operation] — filter by Guest Card's end date and time. Possible values for [operation]: EQ, GT, LT, GTE, LTE. (e.g. api/v1/guestcards?expirationTime%5Bgte%5D=2023-05-10T00:00:00Z) Both ISO 8601 and YYYY-MM-DD hh:mm:ss date-time formats are accepted in requests, but the response is always returned in ISO 8601 format. • <custom_field_name> — e.g. custom field "issueLocation" has been created (e.g. api/v1/guestcards?issueLocation=main%20office) • includeFields — specifies what fields the response should contain (e.g. api/v1/guestcards?includeFields=name,id — only fields "name" and "id" are returned) • excludeFields — specifies what fields should be omitted from the response (e.g. api/v1/guestcards?excludeFields=description — all fields except "description" are returned) • limit — specifies how many entities the response should contain (e.g. api/v1/guestcards?limit=10 — means that only 10 entities are returned). The value of this parameter must be between 0 and 3000 • offset — specifies how many entities should be skipped from the top (e.g. api/v1/guestcards?limit=10&offset=10 — means that 10 records are skipped, and next 10 entities are returned). The value of this parameter must be greater than or equal to 0 • sortBy — specifies sort field (e.g. api/v1/guestcards?sortBy=name — means that entities would be sorted by "name" field) • sortOrder — specifies sort order. Can be either ASC or DESC (e.g. api/v1/guestcards?sortBy=name&sortOrder=DESC — means that entities would be sorted by "name" field in descending order)
Body	
Success Response	Status code: 200 Content-type: application/json Body: <pre>[{ "id": <Number>, "name": <String>, "value": <String>, // card number in hexadecimal format "format": <Enum: [W26, W34, W36, W37, W42, W58, W58DEC]>, "formattedValue": <String>, // formatted representation "startTime": <DateTime>, // guest card validity start "expirationTime": <DateTime>, // guest card validity end "assigned": <Boolean>, // indicates whether the guest card is assigned to a guest or an employee "description": <String> }]</pre>

Error Response	400 (Invalid Request) 401 (Unauthorized) 403 (Permission Denied) 422 (Validation Error) 500 (Internal Server Error)
Example	Request: curl -X GET http://192.168.0.108:9500/api/v1/guestcards -H "Content-Type: application/json" -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlciIsIm" Response: [{ "id": 10, "name": "Guest Badge № 1", "value": "7BB26E", "format": "W26", "formattedValue": "123,45678", "assigned": false } { "id": 11, "name": "Guest Badge № 2", "value": "11BDC1", "format": "W26", "formattedValue": "017,48577", "startTime": "2025-08-10T10:37:42", "expirationTime": "2025-08-11T10:37:42", "assigned": true, "description": "Has RFID sticker replaced" }]

2.16.2. Get guest card by ID

URL Endpoint	api/v1/guestcards/{id}
Method	GET
Header Parameters	Authorization: Bearer <JWT access token>
Path Parameters	id – identifier of a Guest Card
Query Parameters	includeFields – specifies what fields the response should contain (e.g. api/v1/guestcards/8?includeFields=name,id – only fields "name" and "id" are returned) excludeFields – specifies what fields should be omitted from the response (e.g. api/v1/guestcards/8?excludeFields=description – all fields except "description" are returned)
Body	
Success Response	Status code: 200 Content-type: application/json Body: <pre>[{ "id": <Number>, "name": <String>, "value": <String>, // card number in hexadecimal format "format": <Enum: [W26, W34, W36, W37, W42, W58, W58DEC]>, "formattedValue": <String>, // formatted representation "startTime": <DateTime>, // guest card validity start "expirationTime": <DateTime>, // guest card validity end "assigned": <Boolean>, // indicates whether the guest card is assigned to a guest "description": <String> }]</pre>
Error Response	400 (Invalid Request) 401 (Unauthorized) 403 (Permission Denied) 404 (Card With The Specified ID Not Found) 500 (Internal Server Error)
Example	Request: <pre>curl -X GET http://192.168.0.108:9500/api/v1/guestcards/32 -H "Content-Type: application/json" -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilslm"</pre> Response: <pre>[{ "id": 32, "name": "Guest Badge № 14", "value": "11BDC1", "format": "W26", "formattedValue": "017,48577", "startTime": "2025-08-10T10:37:42", "expirationTime": "2025-08-11T10:37:42", "assigned": true, "description": "Has RFID sticker replaced" }]</pre>

2.16.3. Create guest card

As a result, a cardholder of type "Guest badge" is created.

URL Endpoint	api/v1/guestcards
Method	POST
Header Parameters	Authorization: Bearer <JWT access token>
Path Parameters	
Query Parameters	
Body	<pre>{ "name": <String>, "value": <String>, "format": <Enum: [W26, W34, W36, W37, W42, W58, W58DEC]>, "startTime": <DateTime>, "expirationTime": <DateTime>, "description": <String> }</pre> Required fields: name — name of a Guest Card value — a HEX value that must be unique across all QR codes and cards. Comparison includes different Wiegand formats and leading zeros. For example, a W34 card with number AABBCDD is treated as identical to a W58 card with number 00000AABBCDD. Optional fields: format — Wiegand format. Automatically set based on "value" if not specified. If "value" matches multiple Wiegand formats (W36, W37, W42), format W42 is used. If it matches both W58 and W58DEC, format W58 is used. description startTime — if not set, it equals "expirationTime" expirationTime — if not set, it equals "startTime" Both "startTime" and "expirationTime" may be absent (null). In this case, the Guest Card has no expiration limitations. Both ISO 8601 and YYYY-MM-DD hh:mm:ss date-time formats are accepted in requests, but the response is always returned in ISO 8601 format.
Success Response	Status code: 200 Content-type: application/json Body: <pre>[{ "id": <Number>, "name": <String>, "value": <String>, "format": <Enum: [W26, W34, W36, W37, W42, W58, W58DEC]>, "formattedValue": <String>, "startTime": <DateTime>, "expirationTime": <DateTime>, "assigned": <Boolean>, "description": <String> }]</pre>
Error Response	400 (Invalid Request) 401 (Unauthorized) 403 (Permission Denied) 422 (Validation Error) 500 (Internal Server Error)

2.16.4. Update guest card

URL Endpoint	api/v1/guestcards/{id}
Method	PUT
Header Parameters	Authorization: Bearer <JWT access token>
Path Parameters	id – identifier of a Guest Card
Query Parameters	
Body	<pre>{ "name": <String>, "startTime": <DateTime>, "expirationTime": <DateTime>, "description": <String> }</pre> Body contains fields that should be modified. It is possible to modify the following fields: - name - startTime - expirationTime - description <custom_field> Passing an empty string for "startTime" or "expirationTime" removes Guest Card expiration limitations. Limitations can be removed only if the card is currently unassigned (assigned=false). Both ISO 8601 and YYYY-MM-DD hh:mm:ss date-time formats are accepted in requests, but the response is always returned in ISO 8601 format.
Success Response	Status code: 200 Content-type: application/json Body: <pre>[{ "id": <Number>, "name": <String>, "value": <String>, "format": <Enum: [W26, W34, W36, W37, W42, W58, W58DEC]>, "formattedValue": <String>, "startTime": <DateTime>, "expirationTime": <DateTime>, "assigned": <Boolean>, "description": <String> }]</pre>
Error Response	400 (Invalid Request) 401 (Unauthorized) 403 (Permission Denied) 404 (Not Found) 422 (Validation Error) 500 (Internal Server Error)

2.16.6. Create multiple guest cards

URL Endpoint	api/v1/guestcards/create
Method	POST
Header Parameters	Authorization: Bearer <JWT access token>
Path Parameters	
Query Parameters	
Body	<pre>[{ "name": <String>, "value": <String>, "format": <Enum: [W26, W34, W36, W37, W42, W58, W58DEC]>, "startTime": <DateTime>, "expirationTime": <DateTime>, "description": <String> }, ...]</pre> Required fields: name — name of a Guest Card value — a HEX value that must be unique across all QR codes and cards. Comparison includes different Wiegand formats and leading zeros. For example, a W34 card with number AABBCDD is treated as identical to a W58 card with number 00000AABBCDD. Optional fields: format — Wiegand format. Automatically resolved based on "value" if not specified. If "value" matches multiple Wiegand formats (W36, W37, W42), format W42 is used. If it matches both W58 and W58DEC, format W58 is used. description startTime — if not set, it equals "expirationTime" expirationTime — if not set, it equals "startTime" Both "startTime" and "expirationTime" may be absent (null). In this case, the Guest Card has no expiration limitations. Both ISO 8601 and YYYY-MM-DD hh:mm:ss date-time formats are accepted in requests, but the response is always returned in ISO 8601 format.
Success Response	Status code: 200 Content-type: application/json Body: <pre>[{ "id": <Number>, "name": <String>, "value": <String>, "format": <Enum: [W26, W34, W36, W37, W42, W58, W58DEC]>, "formattedValue": <String>, "startTime": <DateTime>, "expirationTime": <DateTime>, "assigned": <Boolean>, "description": <String> }, ...]</pre>
Error Response	400 (Invalid Request) 401 (Unauthorized) 403 (Permission Denied) 422 (Validation Error) 500 (Internal Server Error)

2.16.7. Update multiple guest cards

URL Endpoint	api/v1/guestcards/update
Method	POST
Header Parameters	Authorization: Bearer <JWT access token>
Path Parameters	id — identifier of a Guest Card
Query Parameters	
Body	<pre>[{ "id": <Number>, "name": <String>, "startTime": <DateTime>, "expirationTime": <DateTime>, "description": <String> }, ...]</pre> Body contains fields that should be modified. It is possible to modify the following fields: - name - startTime - expirationTime - description <custom_field> Passing an empty string for "startTime" or "expirationTime" removes Guest Card expiration limitations. Limitations can be removed only if the card is currently unassigned (assigned=false). Both ISO 8601 and YYYY-MM-DD hh:mm:ss date-time formats are accepted in requests, but the response is always returned in ISO 8601 format.
Success Response	Status code: 200 Content-type: application/json Body: <pre>[{ "id": <Number>, "name": <String>, "value": <String>, "format": <Enum: [W26, W34, W36, W37, W42, W58, W58DEC]>, "formattedValue": <String>, "startTime": <DateTime>, "expirationTime": <DateTime>, "assigned": <Boolean>, "description": <String> }, ...]</pre>
Error Response	400 (Invalid Request) 401 (Unauthorized) 403 (Permission Denied) 404 (Not Found) 422 (Validation Error) 500 (Internal Server Error)

Example	Request: <pre>curl -X POST http://192.168.0.108:9500/api/v1/guestcards/update -H "Content-Type: application/json" -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilslm" -d [{"id": 12, "name": "Card 2 (RFID)"}, {"id": 13, "name": "Card 3 (RFID)"}]</pre> Response: <pre>[{ "id": 12, "name": "Card 2 (RFID)", "value": "AABBCCDD", "format": "W34", "formattedValue": "AABBCCDD", "assigned": false }, { "id": 13, "name": "Card 3 (RFID)", "value": "E6D5F6", "format": "W26", "formattedValue": "230,54774", "startTime": "2025-09-29T00:00:00", "expirationTime": "2025-09-29T23:59:00", "assigned": true }]</pre>
---------	---

2.16.8. Delete multiple guest cards

A Guest Card can be deleted only if it is not assigned. Therefore, a card to be deleted must be unassigned first.

URL Endpoint	api/v1/guestcards/delete
Method	POST
Header Parameters	Authorization: Bearer <JWT access token> Content-Type: application/json
Path Parameters	
Query Parameters	
Body	<pre>[<Number>, ...]</pre> In the request body, provide a list of Guests Cards IDs.
Success Response	Status code: 204 No Content
Error Response	<u>400 (Invalid Request)</u> <u>401 (Unauthorized)</u> <u>403 (Permission Denied)</u> <u>404 (Not Found)</u> <u>422 (Validation Error)</u> <u>500 (Internal Server Error)</u>
Example	Request: <pre>curl -X POST http://192.168.0.108:9500/api/v1/guestcards/delete -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilslm" -d '[11, 12]'</pre> Response: 204 No Content

2.17. Guest QR codes

2.17.1. Get guest QR code by ID

URL Endpoint	api/v1/guests/{id}/QR
Method	GET
Header Parameters	Authorization: Bearer <JWT access token>
Path Parameters	id — identifier of a Guest
Query Parameters	
Body	
Success Response	Status code: 200 Content-type: application/json Body: <pre>{ "qrValue": <String>, // QR code number in hexadecimal format "format": <Enum: [W26, W34, W42, W58]>, "formattedValue": <String>, // formatted representation "encodedValue": <String>, // encoded value (system-specific) "image": <String> // QR code image (Base64) }</pre>
Error Response	400 (Invalid Request) 401 (Unauthorized) 403 (Permission Denied) 404 (QR With the Specified Guest ID Not Found) 500 (Internal Server Error)
Example	Request: <pre>curl -X GET http://192.168.0.108:9500/api/v1/guests/8/QR -H "Content-Type: application/json" -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJyYyJlIjojVXNlcilslm"</pre> Response: <pre>{ "qrValue": "E6D5F6", "format": "W26", "formattedValue": "230,54774", "encodedValue": "15128054", "image": "iVBORw0KGgoAAAANSUgAAAPoAAAD" }</pre>

2.17.2. Create guests with QR codes

You can use this endpoint only if the option "Do not permit to issue guest badges if the application No. is missing" is **disabled** in the "Client" tool (under "File – Settings – Guest badges").

Issuing a guest QR Code automatically grants access to all access points. Guests' access rights must be managed through [access rules](#).

URL Endpoint	api/v1/guests/QR/assign
Method	POST
Header Parameters	Authorization: Bearer <JWT access token> Content-Type: application/json
Path Parameters	
Query Parameters	autogenerate – indicates whether "qrValue" should be generated automatically (e.g. api/v1/guests/QR/assign?autogenerate=true) Possible values: - true – QR code is generated according to the settings configured in the "Client" tool (under "File – Settings – PIN and QR codes") - false (default) – "qrValue" must be provided in the request body
Body	<pre>[{ "qrValue": <String>, "guestInfo": { "name": <String>, "birthDate": <Date>, "address": <String>, "lpNumber": <String>, "docType": <String>, "docSeries": <String>, "docNumber": <String>, "hostEmployeeId": <Number>, "wholIssued": <String>, "email": <String>, "issuedDate": <Date>, "photo": <String> }, "startTime": <DateTime>, "expirationTime": <DateTime>, "description": <String> }, ...]</pre> Required fields: name docType – only active document types can be used (i.e., those with the "Show in document types" option enabled in the "Client" tool under "File – Settings – Documents") Optional fields: qrValue – a HEX value (W26, W34, W42, W58) that must be unique across all QR codes and cards. Comparison includes different Wiegand formats and leading zeros. For example, a W34 card with number AABBCDD is treated as identical to a W58 card with number000000AABBCDD startTime – if not set, the current date and time are used expirationTime – if not set, it is calculated as "startTime" plus the "Default guest badge expiration time" value configured in the "Client" tool (under "File – Settings – Guest badges") Both ISO 8601 and YYYY-MM-DD hh:mm:ss date-time formats are accepted in requests, but the response is always returned in ISO 8601 format.

Success Response	Status code: 200 Content-type: application/json Body: <pre>[{ "id": <Number>, "qrValue": <String>, "guestInfo": { "name": <String>, "birthDate": <Date>, "address": <String>, "lpNumber": <String>, "docType": <String>, "docSeries": <String>, "docNumber": <String>, "hostEmployeeId": <Number>, "wholIssued": <String>, "email": <String>, "issuedDate": <Date>, "photo": <String> }, "startTime": <DateTime>, "expirationTime": <DateTime>, "description": <String> }, ...]</pre>
Error Response	400 (Invalid Request) 401 (Unauthorized) 403 (Permission Denied) 422 (Validation Error) 500 (Internal Server Error)
Example	Request: <pre>curl -X POST http://192.168.0.108:9500/api/v1/guests/QR/assign?autogenerate=true -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilsm" -d '[{ "guestInfo": { "name": "Jack Dawson", "docType": "PASSPORT_UZ" } }]'</pre> Response: <pre>{ "id": 6, "qrValue": "A8907A", "guestInfo": { "name": "Jack Dawson", "docType": "PASSPORT_UZ" }, "startTime": "2025-10-10T13:49:03.0319143", "expirationTime": "2025-10-10T20:49:03.0319143" }</pre>

2.17.3. Update guest QR code

URL Endpoint	api/v1/guests/{id}/QR
Method	PUT
Header Parameters	Authorization: Bearer <JWT access token> Content-Type: application/json
Path Parameters	autogenerate — indicates whether "qrValue" should be generated automatically or taken from the value provided in the request body (e.g. api/v1/guests/8/QR?autogenerate=true) Possible values: - true — QR code is generated according to the settings configured in the "Client" tool (under "File — Settings — PIN and QR codes"), and any values provided by the user are ignored - false — "qrValue" and "format" must be provided in the request body
Query Parameters	
Body	<pre>{ "qrValue": <String>, "format": <Enum: [W26, W34, W42, W58]> }</pre> Body contains fields that should be modified. It is possible to modify the following fields: qrValue — a HEX value that must be unique across all QR codes and cards. Comparison includes different Wiegand formats and leading zeros. For example, a W34 card with number AABBCDD is treated as identical to a W58 card with number 000000AABBCDD format — Wiegand format. Possible values: W26, W34, W42, W58. Automatically set based on "qrValue" if not specified. Updating "format" without specifying "qrValue" is not allowed
Success Response	Status code: 200 Content-type: application/json Body: <pre>{ "qrValue": <String>, "format": <Enum: [W26, W34, W42, W58]>, "formattedValue": <String>, "encodedValue": <String>, "image": <String> }</pre>
Error Response	400 (Invalid Request) 401 (Unauthorized) 403 (Permission Denied) 404 (Not Found) 422 (Validation Error) 500 (Internal Server Error)
Example	Request: <pre>curl -X PUT http://192.168.0.108:9500/api/v1/guests/6/QR?autogenerate=false' -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilsm"-d '{"qrValue": "E27733AB"}'</pre> Response: <pre>{ "qrValue": "E27733AB", "format": "W34", "formattedValue": "E27733AB", "encodedValue": "E27733AB", "image": "iVBORw0KGgoAAAANSUhhEUgAAAPoAAA" }</pre>

2.18. Employee virtual identifiers (mobile ID, PIN, QR code)

2.18.1. Overview

These endpoints are designed to manage virtual identifiers: mobile ID, PIN, and QR code.

Virtual identifiers can be assigned to employees only. An employee can have only one of each type. The total number of identifiers per employee, including cards, cannot exceed 5.

To retrieve virtual identifiers, use the [GET /api/v1/employees](#) and [GET /api/v1/employees/{id}](#) endpoints.

To manage verification PIN codes for employees and official vehicles (used in scenarios such as "Card + PIN" and others), use the [Employees](#) and [Official vehicles](#) endpoints.

To manage QR codes for guests, use the [Guest QR codes](#) endpoints.

2.18.2. Assign virtual identifier to employee

An employee can have only one PIN, one QR code, and one mobile ID.

To issue a mobile ID, working with mobile identifiers must be enabled in the Client tool (under File — Settings — Contactless identification — Mobile identification), and the reader must be programmed for the same mode. The mode determines how the identifier reaches the employee's smartphone:

- Base mode — the reader responds to any smartphone with the Sigur Access application installed and running. The "value" must be taken from the application (see the "value" field description below). No further steps are required.
- Default mode — the reader responds only to smartphones that received the identifier via an invitation link. After this request, obtain the link with [GET api/v1/employees/{id}/identifiers/mobile/link](#) and deliver it to the employee. The identifier is added to the Sigur Access application when the employee opens the link.

URL Endpoint	api/v1/employees/{id}/identifiers
Method	POST
Header Parameters	Authorization: Bearer <JWT access token> Content-Type: application/json
Path Parameters	• id — employee identifier
Query Parameters	• autogenerate — indicates whether "value" and "formattedValue" should be generated automatically (e.g. api/v1/employees/8/identifiers?autogenerate=false) Possible values: • true — the identifier is generated automatically: • for PIN and QR codes — according to the settings configured in the Client tool (under File — Settings — PIN and QR codes); • for mobile IDs — in Default mode only. In Base mode the "value" must match the ID taken from the application and cannot be generated automatically. • false (default) — either "value" or "formattedValue" must be provided in the request body • replace — indicates whether to replace an existing virtual identifier with the new one provided in the request (e.g. api/v1/employees/8/identifiers?replace=true) Possible values: • true — if the virtual identifier already exists, it will be replaced • false (default) — if the virtual identifier already exists, a 422 error is returned

Body	<pre>{ "type": <Enum: [QR, PIN, MOBILE_ID]>, "format": <Enum: [W26, W34, W42, W58]>, "value": <String>, "formattedValue": <String>, "startTime": <DateTime>, "expirationTime": <DateTime> }</pre> Required fields: type — determines the type of a virtual identifier to be created value, formattedValue, or both — required when "autogenerate" is false or omitted. If only one is provided, the other is filled in automatically; if both are provided, they must match each other Optional fields: format — Wiegand format. Automatically set based on "value" or "formattedValue" if not specified. Possible values: - for QR codes — W26, W34, W42, W58 - for mobile IDs — W26, W34, W58 value — a sequence of hexadecimal digits (applies to any identifier type). The number of bytes depends on the identifier type: - for QR codes and mobile IDs — must match the Wiegand format: W26 — 3 bytes, W34 — 4, W42 — 5, W58 — 7 - for PIN — always 3 bytes, padded with leading zeros if needed For a mobile ID issued in Base mode, "value" must be the last bytes of the ID copied from the Sigur Access application (the ID is shown in the application under "About"). For example, for the ID 6196540B-05C7-401D-8CFF-37E295-E635C4 and the W58 format, send the last 7 bytes: FF37E295E635C4. formattedValue: - for PIN — a sequence of decimal digits (4–7 digits, 1000–9999999). The number of digits must comply with the settings configured in the Client tool (under File — Settings — PIN and QR codes). PINs with leading zeros are not allowed - for W26 format — facility format (e.g., "123,12345") - in all other cases — equals "value" startTime expirationTime Both "startTime" and "expirationTime" may be specified or omitted (or passed as null). Providing "startTime" without "expirationTime" is not allowed. If "expirationTime" is specified but "startTime" is not, "startTime" is set to the current date and time. Both ISO 8601 and YYYY-MM-DD hh:mm:ss date-time formats are accepted in requests.
Success Response	Status code: 200 Content-type: application/json Body: <pre>{ "type": <Enum: [QR, PIN, MOBILE_ID]>, "format": <Enum: [W26, W34, W42, W58]>, // QR codes and mobile IDs only "value": <String>, "formattedValue": <String>, "encodedValue": <String>, // string encoded in the QR code, QR codes only "image": <String>, // base64 string, QR codes only "startTime": <DateTime>, "expirationTime": <DateTime> }</pre> "encodedValue" may differ from "value". Both "encodedValue" and "image" are read-only.
Error Response	400 (Invalid Request) 401 (Unauthorized) 403 (Access Denied) 422 (Validation Error) 500 (Internal Server Error) 503 (Service Unavailable)

Example 1	Request: <pre>curl -X 'POST' \ 'http://192.168.0.108:9500/api/v1/employees/7/identifiers' \ -H 'Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ0eXNlIjoiaWVXNlciIsIm' \ -H 'Content-Type: application/json' \ -d '{"type": "PIN", "value": "0010E1", "formattedValue": "4321", "startTime": "2025-10-10T08:48:07.725Z", "expirationTime": "2025-10-12T08:48:07.725Z"}'</pre> Response: <pre>{ "type": "PIN", "value": "0010E1", "formattedValue": "4321", "startTime": "2025-10-10T08:48:07.725", "expirationTime": "2025-10-12T08:48:07.725" }</pre>
Example 2	Request: <pre>curl -X 'POST' \ 'http://192.168.0.108:9500/api/v1/employees/7/identifiers' \ -H 'Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ0eXNlIjoiaWVXNlciIsIm' \ -H 'Content-Type: application/json' \ -d '{"type": "QR", "value": "AABBCC"}'</pre> Response: <pre>{ "type": "QR", "format": "W26", "value": "AABBCC", "formattedValue": "170,48076", "encodedValue": "11189196", "image": "iVBORw0KGgoAAAANSUgAAAPoAAD6AQAAAACgl2eQAAAAzE" }</pre> Note: The example image string above is truncated for readability.
Example 3	Request: <pre>curl -X 'POST' \ 'http://192.168.0.108:9500/api/v1/employees/7/identifiers?autogenerate=true' \ -H 'Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ0eXNlIjoiaWVXNlciIsIm' \ -H 'Content-Type: application/json' \ -d '{ "type": "MOBILE_ID" }'</pre> Response: <pre>{ "type": "MOBILE_ID", "format": "W26", "value": "A7EC3E", "formattedValue": "167,60478" }</pre>

2.18.3. Update start and expiration time for virtual identifiers

URL Endpoint	api/v1/employees/{id}/identifiers
Method	PATCH
Header Parameters	Authorization: Bearer <JWT access token> Content-Type: application/json
Path Parameters	id – employee identifier
Query Parameters	
Body	<pre>[{ "type": <Enum: [QR, PIN, MOBILE_ID]>, "format": <Enum: [W26, W34, W42, W58]>, "value": <String>, "formattedValue": <String>, "startTime": <DateTime>, "expirationTime": <DateTime> }, ...]</pre> An array of the employee's virtual identifiers can be provided. Only the startTime and expirationTime values of the specified identifiers are modified. Each entry identifies an existing identifier of the employee: the identifier is matched by "type" together with "value" or "formattedValue" (one of the two is enough); if both are provided, both must match the existing identifier. If "format" is also provided, it must match as well. If any entry does not match an existing virtual identifier of the employee, the request fails and none of the identifiers are updated. Required fields: type - either value or formattedValue Optional fields: format – Wiegand format (only for QR codes and mobile IDs) startTime expirationTime Both "startTime" and "expirationTime" may be set together, omitted (or passed as null), or cleared together (by passing empty strings). Omitting a field or passing null leaves its current value unchanged. If neither field is currently set, providing only one of them is not allowed – both must be set together. If both are already set, updating a single field is allowed. Clearing only one field is not allowed. Both ISO 8601 and YYYY-MM-DD hh:mm:ss date-time formats are accepted in requests.
Success Response	Status code: 204 No Content
Error Response	401 (Unauthorized) 403 (Access Denied) 422 (Validation Error) 500 (Internal Server Error)

Example	Request: <pre>curl -X 'PATCH' \ 'http://192.168.0.108:9500/api/v1/employees/7/identifiers' \ -H 'Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlciIsIm' \ -H 'Content-Type: application/json' \ -d '[{ "type": "QR", "value": "AABBCC", "startTime": "2026-09-01T08:00:00", "expirationTime": "2026-09-01T22:00:00" }, { "type": "PIN", "value": "0010E1", "startTime": "", "expirationTime": "" }]'</pre> Response: 204 No Content
---------	--

2.18.4. Remove virtual identifier

URL Endpoint	api/v1/employees/{id}/identifiers/delete
Method	POST
Header Parameters	Authorization: Bearer <JWT access token> Content-Type: application/json
Path Parameters	id — employee identifier
Query Parameters	
Body	<pre>{ "type": <Enum: [QR, PIN, MOBILE_ID]>, "format": <Enum: [W26, W34, W42, W58]>, "value": <String>, "formattedValue": <String> }</pre> The identifier to be removed is matched by "type" together with "value" or "formattedValue" (one of the two is enough); if both are provided, both must match the existing identifier. If "format" is also provided, it must match as well. If no matching identifier is found, the request fails. Required fields: type - either value or formattedValue Optional fields: format — Wiegand format (only for QR codes and mobile IDs)
Success Response	Status code: 204 No Content
Error Response	<u>401 (Unauthorized)</u> <u>403 (Access Denied)</u> <u>422 (Validation Error)</u> <u>500 (Internal Server Error)</u>
Example	Request: <pre>curl -X 'POST' \ 'http://192.168.0.108:9500/api/v1/employees/7/identifiers/delete' \ -H 'Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilzIm' \ -H 'Content-Type: application/json' \ -d '{ "type": "QR", "value": "AABBCC" }'</pre> Response: 204 No Content

2.19. Roles

2.19.1. Get all roles

URL Endpoint	api/v1/roles
Method	GET
Header Parameters	Authorization: Bearer <JWT access token>
Path Parameters	
Query Parameters	name — filter by Role name with condition "starts with" (e.g. api/v1/roles?name=Operator) includeArchived — specifies whether archived roles should be included in the response (e.g. api/v1/roles?includeArchived=true) limit — specifies how many entities the response should contain (e.g. api/v1/roles?limit=10 — means that only 10 entities are returned). The value of this parameter must be between 0 and 3000 offset — specifies how many entities should be skipped from the top (e.g. api/v1/roles?limit=10&offset=10 — means that 10 records are skipped, and next 10 entities are returned). The value of this parameter must be greater than or equal to 0
Body	
Success Response	Status code: 200 Content-type: application/json Body: <pre>[{ "id": <Number>, "name": <String>, "status": <Enum: [ACTIVE, ARCHIVED]>, "permissions": [// all permissions granted to the role { "accessible": <Boolean>,// indicates whether access is granted "name": <String>// name of the entity to which access is granted }, ...] }, ...]</pre>
Error Response	400 (Invalid Request) 401 (Unauthorized) 403 (Access Denied) 422 (Validation Error) 500 (Internal Server Error)

Example	Request: <pre>curl -X GET http://192.168.0.108:9500/api/v1/roles -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilslm"</pre> Response: <pre>[{ "id": 1, "name": "Operator", "status": "ACTIVE", "permissions": [{ "accessible": false, "name": "USER_T_HW" }, { "accessible": false, "name": "USER_T_PERSONAL_BADGES_EDIT" }, ... { "accessible": false, "name": "USER_T_MON_CNTLAP" }] }]</pre> Note: The response contains a full list of permissions. The example above is truncated for readability.
---------	---

2.19.2. Get role by ID

URL Endpoint	api/v1/roles/{id}
Method	GET
Header Parameters	Authorization: Bearer <JWT access token>
Path Parameters	id — Role identifier
Query Parameters	
Body	
Success Response	Status code: 200 Content-type: application/json Body: <pre>[{ "id": <Number>, "name": <String>, "status": <Enum: [ACTIVE, ARCHIVED]>, "permissions": [// all permissions granted to the role { "accessible": <Boolean>, // indicates whether access is granted "name": <String> // name of the entity to which access is granted }, ...] }]</pre>
Error Response	400 (Invalid Request) 401 (Unauthorized) 403 (Access Denied) 404 (Role with the specified ID not found) 500 (Internal Server Error)
Example	Request: <pre>curl -X GET http://192.168.0.108:9500/api/v1/roles/1 -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilsm"</pre> Response: <pre>[{ "id": 1, "name": "Operator", "status": "ACTIVE", "permissions": [{ "accessible": false, "name": "USER_T_HW" }, { "accessible": false, "name": "USER_T_PERSONAL_BADGES_EDIT" }, ... { "accessible": false, "name": "USER_T_MON_CNTLAP" }] }]</pre> Note: The response contains a full list of permissions. The example above is truncated for readability.

2.19.3. Assign role to user

URL Endpoint	api/v1/roles/{roleId}/assign-to/{userId}
Method	POST
Header Parameters	Authorization: Bearer <JWT access token>
Path Parameters	▪ roleId – Role identifier ▪ userId – User identifier
Query Parameters	
Body	
Success Response	Status code: 204 No Content
Error Response	<u>400 (Invalid Request)</u> <u>401 (Unauthorized)</u> <u>403 (Access Denied)</u> <u>404 (User or Role with the specified ID not found)</u> <u>500 (Internal Server Error)</u>
Example	Request: <pre>curl -X POST http://192.168.0.108:9500/api/v1/roles/5/assign-to/10 -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1YyIjoiVXNlciIsIm"</pre> Response: 204 No Content

2.19.4. Unassign role from user

URL Endpoint	/api/v1/roles/remove-from/{userId}
Method	DELETE
Header Parameters	Authorization: Bearer <JWT access token>
Path Parameters	userId – User identifier
Query Parameters	
Body	
Success Response	Status code: 204 No Content
Error Response	<u>401 (Unauthorized)</u> <u>403 (Access Denied)</u> <u>404 (User with the specified ID not found)</u> <u>500 (Internal Server Error)</u>
Example	Request: <pre>curl -X POST http://192.168.0.108:9500/api/v1/roles/remove-from/10 -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1IjoiVXNlcilslm"</pre> Response: 204 No Content

3. Response error codes

3.1. 400 Invalid Request

Status code: 400
Content-type: application/json
Body:

```
{
  "status": 400,
  "errors": [
    <Error description>
  ]
}
```

3.2. 401 Unauthorized

Status code: 401
Content-type: application/json
Body:

```
{
  "status": 401,
  "errors": [
    <Error description>
  ]
}
```

3.3. 403 Forbidden

Status code: 403
Content-type: application/json
Body:

```
{
  "status": 403,
  "errors": [
    <Error description>
  ]
}
```

3.4. 404 Not Found

Status code: 404
Content-type: application/json
Body:

```
{
  "status": 404,
  "errors": [
    <Error description>
  ]
}
```

3.5. 422 Unprocessable Entity

Status code: 422
Content-type: application/json
Body:

```
{
  "status": 422,
  "errors": [
    <Error description>
  ]
}
```

3.6. 500 Internal Server Error

Status code: 500
Content-type: application/json
Body:

```
{
  "status": 500,
  "errors": [
    <Error description>
  ]
}
```

3.7. 503 Service Unavailable

Status code: 503
Content-type: application/json
Body:

```
{
  "status": 503,
  "errors": [
    <Error description>
  ]
}
```

4. Contacts

For any inquiries or assistance, please contact us using the provided information.

Website: www.sigur.com

General Inquiries: info@sigur.com

Technical Support: support@sigur.com