# Mopuan Room QR Changed

## 1. What this service is for

The OpSense portal shows each room's access method (QR, RFID or mobile) and lets staff view and download room QR codes. It reads this data from your `room-qr/{propertyId}` service and keeps a cached copy so pages load quickly.

**Mopuan Room QR Changed** tells us that the room data for one or more buildings has changed. When you call it, we drop our cached copy for those buildings. The next time anyone in the portal views those rooms, we fetch the current data from you.

You do not send the new QR codes or room data to this service. You only tell us **which buildings** changed, and we fetch the data ourselves.

> If you do not call this service after a change, the portal can keep showing the old QR code for up to **1 hour**.

## 2. When to call it

Call it every time any of the following changes for a building, **after the change has been saved** on your side:

- A room's QR code is created, regenerated or removed.
- A room's access type changes (for example from RFID to QR, or QR to mobile).
- A room is added to or removed from the building.
- Any bulk operation that does one of the above for many rooms.

If you call us before your change is saved, we may fetch the old data again and cache it.

When several buildings change at once, send them all in one call (see `property_ids` below).

## 3. Authentication

Send the company's `api_key` as a normal request parameter, the same way as for the other OpSense services: as a POST field. Do not send it as an HTTP header.

The key decides which company the call is for. You can only mark buildings of that company as changed.

| Situation | Response |
|---|---|
| `api_key` missing | `error_code 10015` |
| `api_key` invalid or inactive | `error_code 10016` |

Every response is wrapped the same way:

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

## 4. Request

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

Send the parameters as **form fields**. A JSON request body is not read.

| Field | Required | Notes |
|---|---|---|
| `api_key` | **Required** | See Authentication. |
| `property_ids` | **Required, unless `property_id` is sent** | The buildings that changed. A comma-separated list (`12,15,20`) or a form array (`property_ids[]=12&property_ids[]=15`). Maximum 500 per call. |
| `property_id` | Optional | A single building. Use either `property_id` or `property_ids`. If both are sent, `property_ids` is used. |

The building id is the same id you use in `room-qr/{propertyId}`.

Duplicate ids are ignored.

### Examples

One building:

```
curl -X POST "https://<api-host>/v1/SRVMopuanRoomQrChanged" \ -d "api_key=<api_key>" \ -d
"property_id=12"
```

Several buildings:

```
curl -X POST "https://<api-host>/v1/SRVMopuanRoomQrChanged" \ -d "api_key=<api_key>" \ -d
"property_ids=12,15,20"
```

## 5. Response

### Success

```
{ "result": { "result": "success", "updated": [12, 15], "unknown": [20] } }
```

| Field | Description |
|---|---|
| `result` | `"success"` |
| `updated` | The buildings whose cached data we dropped. |
| `unknown` | Ids we ignored because they are not buildings of the company that owns the `api_key`. |

A call that only contains unknown ids still returns `success`, with an empty `updated` list. Retrying it does not change the outcome, so do not retry it. Check that you are using the right `api_key` for that building.

Calling the service again for the same building is safe. It has no effect beyond refreshing the data one more time.

### Errors

| error_code | Meaning | Retry? |
|---|---|---|
| 10015 | `api_key` missing | No. Fix the request. |
| 10016 | `api_key` invalid or inactive | No. Check the key. |
| 10020 | No valid building ids were sent, or more than 500 | No. Fix the request. |
| 10099 | The request could not be completed on our side | Yes |

```
{"result": {"error_code": "10020", "error_text": "property_id or property_ids is required."}}
```

## 6. Retries

If the call fails because of a network error, a timeout, an HTTP status other than 200, or `error_code 10099`, retry it. We suggest waiting 5 seconds, then 30 seconds, then 2 minutes, then 10 minutes.

You do not need to keep retrying for longer than 1 hour. After 1 hour our cached copy expires and we fetch the current data anyway.

## 7. What happens on our side

1. Your call marks the listed buildings as changed.
2. The next request in the portal for one of those buildings sees the mark and fetches `room-qr/{propertyId}` from you again instead of using the cached copy.
3. Pages that are already open pick up the change the next time a user opens a room's QR code.
4. Buildings you did not list keep using their cached copy, so we do not call your service more often than needed.