Dokumentation REST-APIs zum Zugriff auf den Hausmanager für externe Systeme

net-haus GmbH
Software und Service
Arkonastraße 45-49
13189 Berlin

1. Änderungsverzeichnis

Version	Datum	Änderung	Name/Ersteller
1.0	04.04.2024	Version 1	B.Peris (net-haus)
1.1	08.04.2024	Version 1.1	B.Peris (net-haus)

2. Einleitung und Hinweise

Dieses Kapitel beinhaltet allgemeine Hinweise zu den Schnittstellen und ihrer Verwendung.

2.1. Allgemeines

Datumangaben
Im Request wird das Datum im Format YYYY-MM-DD erwartet.

Authentication
Nach dem erfolgreichen Login über die API muss bei jedem weiteren Request der Bearer Token im Header mit angegeben werden.

Return Message im Fehlerfall / Beispielmessage

```
{
    "type": "/processing-error",
    "title": null,
    "status": 400,
    "method": "POST",
    "instance": "/rest/api/address-management/addresses",
    "message": "Invalid value for WageGroup. Value has to be between 1 and 12"
}
```

SWAGGER
Das Dokument gilt immer im Zusammenhang mit der API Doku die unter folgender URL zu erreichen ist:

- {{baseUrl}}/ swagger/ui/index

baseUrl

- {hausmanagerURL}/rest

hausmanagerURL

- Wird vom Projektpartner mitgeteilt

3. APIs im Detail

Die einzelnen APIs werden im Folgenden näher beschrieben. Vor der Benutzung der weiteren APIs muss ein Login durchgeführt werden. Der verwendete Benutzer zum Login muss ein Nutzer des Hausmanagers sein. Dieser wird in der Nutzerverwaltung des Hausmanagers angelegt und administriert.

3.1. Authentication

Über diese API erfolgt der Login zum Hausmanager. Alle API-Requests benötigen einen gültigen API-Token, der nach der Anmeldung übermittelt wird und durch den Bearer Token repräsentiert wird.

Request Method: POST

{{baseUrl}}/api/security/authentication/login

Beipspiel:

{{baseUrl}}/api/security/authentication/login

Body:

```
{
  "UserName": "Username aus dem Hausmanager",
  "Password": "Passwort"
}
```

3.1.1. Eingabeparameter

Name	Datentyp	Länge	Optio-nal	Bemerkung
UserName	String			Username - wird im Hausmanager definiert
Passwort	String			Passwort - wird im Hausmanager generiert

3.1.2. Ausgabeparameter

- Bearer Token

3.2. Adressen

3.2.1. Adressen anlegen

Über diese API können im Hausmanager Adressen angelegt werden. Adressen im Hausmanager haben immer einen Adresstyp, der immer mit angegeben werden muss. Zu Beachten ist, dass nicht jeder Adresstyp alle Felder anbietet. Wenn die Felder gefüllt werden, aber zur Adresse nicht sichtbar sind im Hausmanager, wird das zwar über die API akzeptiert, aber nur die für den Adresstyp gültigen Werte gespeichert.

Request Method: POST

{{baseUrl}}/api/address-management/addresses

Beipspiel:

{{baseUrl}}/api/address-management/addresses

Body:

```
{
    "AddressType": 0,
    "CompanyLabel": "string",
    "Number": "string",
    "Salutation": 0,
    "StaffLabel": "string",
    "FirstName": "string",
    "MiddleName": "string",
    "LastName": "string",
    "FirstName2": "string",
    "MiddleName2": "string",
    "LastName2": "string",
    "Street": "string",
    "ZipCode": "string",
    "City": "string",
    "State": "string",
    "Country": "string",
    "Phone": "string",
    "Phone2": "string",
    "MobilePhone": "string",
    "Fax": "string",
    "Fax2": "string",
    "Pager": "string",
    "Email": "string",
    "Homepage": "string",
    "WageGroup": 0,
    "ServiceType": "string",
    "CustomerExtra": "string",
    "CostCentreDatev": "string",
    "Client": "string",
    "Division": "string",
    "Notes": "string"
}
```

3.2.2. Eingabeparameter

Name	Datentyp	Länge	Pflicht	Beschreibung
AddressType	integer		X	Art der Adresse 0 = Nutzer / Bewohner 1 = Mitarbeiter 2 = Objekt 3 = Eigentümer 4 = Rechnungsadresse 5 = Firma 6 = Tour
CompanyLabel	string			Bezeichnung des Unternehmens
Number	string			Nummer
Salutation	string			Anrede (z. B. Herr, Frau)
StaffLabel	string			Bezeichnung des Mitarbeiters
FirstName	string			Vorname
MiddleName	string			Zweiter Vorname
LastName	string			Nachname
FirstName2	string			Vorname 2
MiddleName2	string			Zweiter Vorname 2
LastName2	string			Nachname 2
Street	string			Straße
ZipCode	string			Postleitzahl
City	string			Stadt
State	string			Bundesland
Country	string			Land
Phone	string			Telefonnummer
Phone2	string			Telefonnummer 2
MobilePhone	string			Mobiltelefonnummer
Fax	string			Faxnummer
Fax2	string			Faxnummer 2
Pager	string			Pager-Nummer
Email	string			E-Mail-Adresse
Homepage	string			Webseite
WageGroup	integer			Lohngruppe (Wert zwischen 1 und 12) oder null
ServiceType	string			Leistungsbereich. Wert aus dem Haus-manager.
CustomerExtra	string			Zusätzliche Kundeninformationen
CostCentreDatev	string			Kostenstellenbezeichnung für Datev
Client	string			Kunde
Division	string			Abteilung
Notes	string			Notizen

3.2.3. Ausgabeparameter

Name	Datentyp	Länge	Bemerkung
Id	string		die interne Adress ID des Hausmanagers
Message	String		Meldung

3.2.4. Adressen aktualisieren

Request Method: PATCH

{{baseUrl}}/api/address-management/addresses/{id}

Beipspiel:

{{baseUrl}}/api/address-management/addresses/10047

Body:

```
{
    "AddressType": 0,
    "CompanyLabel": "string",
    "Number": "string",
    "Salutation": 0,
    "StaffLabel": "string",
    "FirstName": "string",
    "MiddleName": "string",
    "LastName": "string",
    "FirstName2": "string",
    "MiddleName2": "string",
    "LastName2": "string",
    "Street": "string",
    "ZipCode": "string",
    "City": "string",
    "State": "string",
    "Country": "string",
    "Phone": "string",
    "Phone2": "string",
    "MobilePhone": "string",
    "Fax": "string",
    "Fax2": "string",
    "Pager": "string",
    "Email": "string",
    "Homepage": "string",
    "WageGroup": 0,
    "ServiceType": "string",
    "CustomerExtra": "string",
    "CostCentreDatev": "string",
    "Client": "string",
    "Division": "string",
    "Notes": "string"
}
```

3.2.5. Eingabeparameter

Name	Datentyp	Länge	Pflicht	Beschreibung
AddressType	integer		X	Art der Adresse 0 = Nutzer / Bewohner 1 = Mitarbeiter 2 = Objekt 3 = Eigentümer 4 = Rechnungsadresse 5 = Firma 6 = Tour
CompanyLabel	string		x	Bezeichnung des Unternehmens
Number	string			Nummer
Salutation	string			Anrede (z. B. Herr, Frau)
StaffLabel	string			Bezeichnung des Mitarbeiters
FirstName	string			Vorname
MiddleName	string			Zweiter Vorname
LastName	string			Nachname
FirstName2	string			Vorname 2
MiddleName2	string			Zweiter Vorname 2
LastName2	string			Nachname 2
Street	string			Straße
ZipCode	string			Postleitzahl
City	string			Stadt
State	string			Bundesland
Country	string			Land
Phone	string			Telefonnummer
Phone2	string			Telefonnummer 2
MobilePhone	string			Mobiltelefonnummer
Fax	string			Faxnummer
Fax2	string			Faxnummer 2
Pager	string			Pager-Nummer
Email	string			E-Mail-Adresse
Homepage	string			Webseite
WageGroup	integer			Lohngruppe (Wert zwischen 1 und 12) oder null
ServiceType	string			Leistungsbereich. Wert aus dem Haus-manager.
CustomerExtra	string			Zusätzliche Kundeninformationen
CostCentreDatev	string			Kostenstellenbezeichnung für Datev
Client	string			Kunde
Division	string			Abteilung
Notes	string			Notizen

3.2.6. Ausgabeparameter

Name	Datentyp	Länge	Bemerkung
Id	string		die interne Adress ID des Hausmanagers
Message	String		Meldung

3.3. Objekte

Über diese API können im Hausmanager Objekte (Liegenschaften, Gebäude, Etagen, Räume) angelegt, aktualsiert, einzeln oder als Liste abgefragt werden.

3.3.1. Objekte anlegen

Alles, was benötigt wird, ist ein Name ("objectName") und eine Referenz zu einem übergeordneten Objekt. Dies kann entweder der Gruppenname ("groupName") oder ein übergeordnetes Objekt ("parentId") sein. Das Attribut „parentId“ hat immer Priorität. Wenn es angegeben ist, wird der „groupName“ ignoriert, und Das Objekt wird als Kind des Elternobjekts erstellt, falls möglich (Achtung: es ist nicht möglich, Kinder für Räume zu erstellen, da der Raum die tiefstmögliche Ebene des Hausmanagers ist). Wenn der Gruppenname verwendet wird, wird die Gruppe erstellt, wenn diese noch nicht im Hausmanager vorhanden ist. Die Objekte werden darunter als "Liegenschaften" erstellt. Alle anderen Werte sind optional und müssen nicht ausgefüllt werden. Der Standardwert für "isDeactivated" ist false, auch wenn kein Wert dafür angegeben wurde.
Wenn eine Objektadresse ("objectAddress") angegeben ist, erstellt die API eine Objektadresse mit den angegebenen Werten, sofern das neu erstellte Objekt vom Typ Liegenschaft oder Gebäude ist. Die Objektadresse wird in den darunter liegenden Ebenen (Einheut und Räume) ignoriert.
Optional können auch Dokumente mit dem Objekt hochgeladen werden. Ein Dokumentenname und der Inhalt der Datei als Base64-Zeichenfolge sind immer erforderlich.

Request Method: POST

{{baseUrl}}/api/Object

Beipspiel:

{{baseUrl}}/api/Object

Body:


```
{
  "ParentId": 0,
  "GroupName": "_API-Folder",
  "ObjectNumber": "L-000001",
  "ObjectName": "Liegenschaft",
  "IsDeactivated": false,
  "UsageType": "Nutzungsart",
  "Nrf": 0,
  "Bgf": 0,
  "IsVacant": false,
  "CustomField1": "Kundenspezifisches Zusatzfeld 1 in Objektdaten",
  "CustomField2": "Kundenspezifisches Zusatzfeld 2 in Objektdaten",
  "CostCenterNumber": "Nummer der Kostenstelle",
  "CostCenterName": "Name der Kostenstelle",
  "DatevCostCenter": "DATEV Kostenstelle",
  "ObjectForeignKey": "Id aus dem externen Systeme",
  "ObjectAddress": {
    "Zip": "13189",
    "Street": "Arkonastraße 45-49",
```

```
    "City": "Berlin",
    "CountryISO3166": "DE",
    "State": "Deutschlang"
  },
  "Notes": "Notizen zu einem Objekt: Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam",
  "Documents": [
    {
      "FileName": "Bürogebäude.png",
      "ContentAsBase64": "Base64"
    }
  ]
}
```


3.3.1.1. Eingabeparameter

Name	Datentyp	Länge	Pflicht	Beschreibung
ParentId	number			0
GroupName	string			_API-Folder
ObjectNumber	string		**X**	L-000001
ObjectName	string			Liegenschaft
IsDeactivated	boolean			false
UsageType	string			Nutzungsart
Nrf	number			0
Bgf	number			0
IsVacant	boolean			false
CustomField1	string			Kundenspezifisches Zusatzfeld 1 in Objekt-daten
CustomField2	string			Kundenspezifisches Zusatzfeld 2 in Objekt-daten
CostCenterNumber	string			Nummer der Kostenstelle
CostCenterName	string			Name der Kostenstelle
DatevCostCenter	string			DATEV Kostenstelle
ObjectForeignKey	string			Id aus dem externen System
ObjectAddress.Zip	string			13189
ObjectAddress.Street	string			Arkonastraße 45-49
ObjectAddress.City	string			Berlin
ObjectAddress.CountryISO3166	string			DE
ObjectAddress.State	string			Deutschland
Notes	string			Notizen zu einem Objekt: Lorem ipsum do-lor sit amet, consetetur sadipscing elitr, sed diam
Documents[0].FileName	string			Bürogebäude.png
Documents[0].ContentAsBase64	string			Base64

3.3.1.2. Ausgabeparameter

Name	Datentyp	Länge	Bemerkung
id	string	255	die interne Objekt ID des Hausmanagers

3.3.1.3. HTTP Status Codes

Bad Request 400:

- unknown/forbidden fileType (return also a list of supported fileextensions)
- parentId not found
- parent is deactivated
- parent can't have childeren (parent is type room)
- countryISO3166 has no valid value
- ngf/nrf must be positive
- there already exists an object wird the same name(on liegenschaft level)

Unauthorized 401:

- Authentication failed

Not Allowed 403:

- user has no "permission" for object(also can't create childeren)

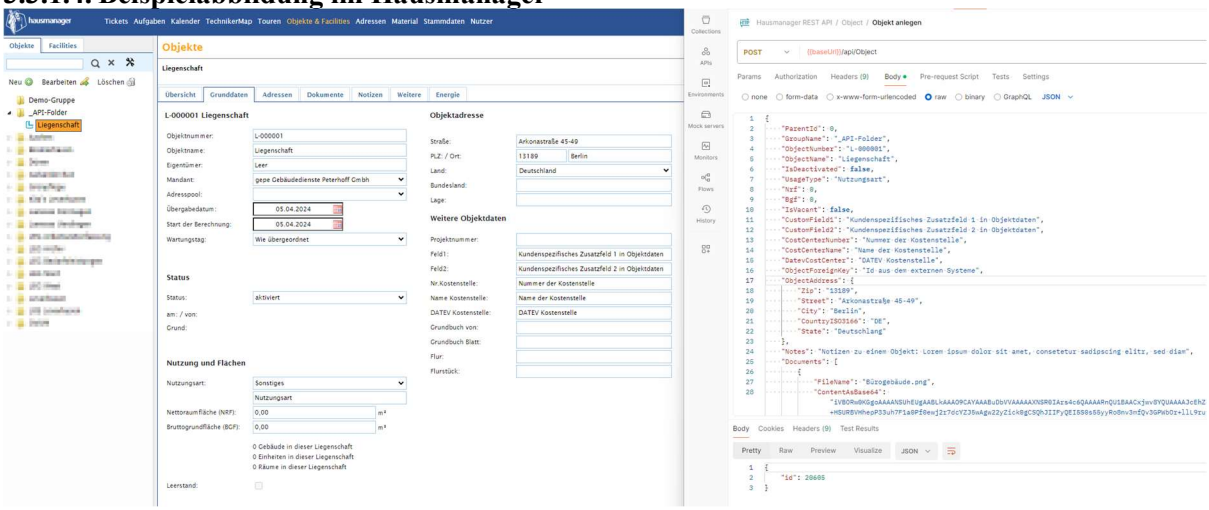
Internal Server Error 500

- Server had error during creation of object

Created 201

- returns Id of created Object

3.3.1.4. Beispielabbildung im Hausmanager



3.3.2. Objekte aktualisieren

Der Request für das Aktualisieren ist ähnlich. Dieser hat jedoch keine „parentId“ oder „groupName“, sondern „Id“, die das einzige obligatorische Feld ist. Alle weiteren Felder sind optional, was bedeutet, dass im Request nur die Felder gesendet werden müssen, die zu aktualisieren sind. Unten sind einige gültige JSONs aufgeführt, die der Aufrufer senden könnte. Die kompliziertesteÄnderung, die der Aufrufer vornehmen kann, ist das Festlegen von "isDeactivated" auf true. Dies bedeutet, dass das Objekt und alle seine Bauteile deaktiviert und alle Aufgaben auf "erledigt" gesetzt werden müssen. Das Verfahren wird auch direkt auf alle seine „Kinder“ angewendet. Das kann eine laufzeitintensive Operation werden.

Request Method: PATCH

{{baseUrl}}/api/object-management/objects/{objectId}

Beipspiel:

{{baseUrl}}/api/object-management/objects/20606

3.3.2.1. Eingabeparameter

Name	Datentyp	Länge	Pflicht	Beschreibung
Id	string		X	Id des Objektes im Hausmanager
ObjectNumber	string			L-000001
ObjectName	string			Liegenschaft
IsDeactivated	boolean			false
UsageType	string			Nutzungsart
Nrf	number			0
Bgf	number			0
IsVacant	boolean			false
CustomField1	string			Kundenspezifisches Zusatzfeld 1 in Objekt-daten
CustomField2	string			Kundenspezifisches Zusatzfeld 2 in Objekt-daten
CostCenterNumber	string			Nummer der Kostenstelle
CostCenterName	string			Name der Kostenstelle
DatevCostCenter	string			DATEV Kostenstelle
ObjectForeignKey	string			Id aus dem externen Systeme
ObjectAddress.Zip	string			13189
ObjectAddress.Street	string			Arkonastraße 45-49
ObjectAddress.City	string			Berlin
ObjectAddress.CountryISO3166	string			DE
ObjectAddress.State	string			Deutschland
Notes	string			Notizen zu einem Objekt: Lorem ipsum do-lor sit amet, consetetur sadipscing elitr, sed diam
Documents[0].FileName	string			Bürogebäude.png
Documents[0].ContentAsBase64	string			Base64

3.3.2.2. Ausgabeparameter / http Status Codes

Bad Request 400:

- Id not found
- unknown fileType
- countryISO3166 has no valid value
- ngf/nrf must be positive
- there already exists an object wird the same name (on liegenschaft level)

Unauthorized 401:

- Authentication failed

Not Allowed 403:

- user has no "permission" for object (also can't create childeren)

Internal Server Error 500

- Server had error during update of object

No Content 204

- on success

3.3.2.3. Sample Requests

Datei hinzufügen

```
{
  "Id": 20606, //Pflicht
  "documents": [
    {
      "fileName": "Document1.pdf",
      "fileContent": "Base64EncodedStringOfDocument1"
    },
    {
      "fileName": "Document2.docx",
      "fileContent": "Base64EncodedStringOfDocument2"
    }
  ]
}
```

Aktualisieren oder hinzufügen einer Objektadresse

```
{
  "Id": 20606, //mandatory
  "objectAddress": {
    "zip": "12345",
    "street": "Sample Street",
    "city": "Sample City",
    "countryISO3166": "ISO 3166 Code",
    "state": "Sample State"
  }
}
```

Weitere

```
{
  "Id": 20606, //mandatory
  "isDeactived": true,
  "usageType": "Other",
  "nrf": 0,
  "bgf": 234.56,
  "customField1": "Custom Field 1 Data",
  "notes": "",
  "objectAddress": {
    "zip": "12345",
    "street": "Sample Street",
    "city": "Sample City",
    "countryISO3166": "ISO 3166 Code",
    "state": "Sample State"
```

```
  }
}
```

3.3.3. Ein Objekt abfragen

Mit dem Get-Endpunkt erhalten Sie alle Informationen zu einem Objekt für die ID, die Sie senden. Dokumente werden jedoch nicht enthalten sein. Es gibt einen optionalen Parameter, der auch alle Kinder des angeforderten Objekts abrufen kann. Die Ergbbisse werden dann als nested object dargestellt.

Request Method: GET

{{baseUrl}}/api/object-management/objects/{objectId}

Beipspiel:

{{baseUrl}}/api/object-management/objects/20606

{{baseUrl}}/api/object-management/objects/20606?withChildren=1

3.3.3.1. Eingabeparameter

Name	Datentyp	Länge	Pflicht	Beschreibung
objectId	long			Objekt Id des Hausmanagers
withChildren	integer			Mit Kindern = 1 oder Ohne dann leer oder 0

3.3.3.2. Ausgabe / http Status Codes


```
[
  {
    "ObjectId": 20606,
    "GroupName": "_API-Folder",
    "ObjectNumber": "L-000002",
    "ObjectName": "Liegenschaft 02",
    "IsDeactivated": false,
    "UsageType": "Nutzungsart",
    "Nrf": 0.0,
    "Bgf": 0.0,
    "IsVacant": false,
    "CustomField1": "Kundenspezifisches Zusatzfeld 1 in Objektdaten",
    "CustomField2": "Kundenspezifisches Zusatzfeld 2 in Objektdaten",
    "CostCenterNumber": "Nummer der Kostenstelle",
    "CostCenterName": "Name der Kostenstelle",
    "DatevCostCenter": "DATEV Kostenstelle",
    "ObjectForeignKey": "Id aus dem externen Systeme",
    "ObjectAddress": {
      "Zip": "13189",
      "Street": "Arkonastraße 45-49",
      "City": "Berlin",
      "CountryISO3166": null,
      "State": null
    },
    "Notes": "Notizen zu einem Objekt: Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam",
    "Children": [
```

```
        {
          "ObjectId": 20607,
          "GroupName": "_API-Folder",
          "ObjectNumber": "G-000001",
          "ObjectName": "Gebäude 01",
          "IsDeactivated": false,
          "UsageType": "Nutzungsart",
          "Nrf": 0.0,
          "Bgf": 0.0,
          "IsVacant": false,
          "CustomField1": "Kundenspezifisches Zusatzfeld 1 in Objektdaten",
          "CustomField2": "Kundenspezifisches Zusatzfeld 2 in Objektdaten",
          "CostCenterNumber": "Nummer der Kostenstelle",
          "CostCenterName": "Name der Kostenstelle",
          "DatevCostCenter": "DATEV Kostenstelle",
          "ObjectForeignKey": "Id aus dem externen Systeme",
          "ObjectAddress": {
            "Zip": "13189",
            "Street": "Arkonastraße 45-49",
            "City": "Berlin",
            "CountryISO3166": null,
            "State": null
          },
          "Notes": "Notizen zu einem Objekt: Lorem ipsum dolor sit amet, consetetur sadipscing elitr,
sed diam",
          "Children": null
        }
      ]
    }
]
```


3.3.4. Eine Liste von Objekten abfragen

Der letzte Endpunkt ruft eine Liste aller Objekte ab, für die der Benutzer (siehe Authentication) Berechtigungen hat. Er enthält immer alle Kinder. Es könnte sich um eine große Menge an Daten handeln. Um alle Objekte einer Objektgruppe abzufragen kann optioanl der Parameter „groupName“ angegeben werden

Request Method: GET

{{baseUrl}}/api/object-management/objects

Beipspiel:

{{baseUrl}}/api/object-management/objects
{{baseUrl}}/api/object-management/objects?groupName=_API-Folder

3.3.4.1. Eingabeparameter

Name	Datentyp	Länge	Pflicht	Beschreibung
groupName	string			Objektgruppe des Hausmanagers

3.3.4.2. Ausgabe / http Status Codes

Bad Request 400:
- Id not found

Unauthorized 401:
- Authentication failed

Not Allowed 403:
- user has no "permission" for object

Internal Server Error 500
- Server had error during update of object

OK 200
- on success with objectList


```
[
  {
    "ObjectId": 20605,
    "GroupName": "_API-Folder",
    "ObjectNumber": "20605",
    "ObjectName": "Liegenschaft",
    "IsDeactivated": false,
    "UsageType": "Nutzungsart",
    "Nrf": 0.0,
    "Bgf": 0.0,
    "IsVacant": false,
    "CustomField1": "Kundenspezifisches Zusatzfeld 1 in Objektdaten",
    "CustomField2": "Kundenspezifisches Zusatzfeld 2 in Objektdaten",
    "CostCenterNumber": "Nummer der Kostenstelle",
    "CostCenterName": "Name der Kostenstelle",
    "DatevCostCenter": "DATEV Kostenstelle",
    "ObjectForeignKey": "Id aus dem externen Systeme",
    "ObjectAddress": {
      "Zip": "13189",
      "Street": "Arkonastraße 45",
      "City": "Berlin",
      "CountryISO3166": null,
      "State": null
    },
    "Notes": "Notizen zu einem Objekt: Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam",
    "Children": null
  },
  {
    "ObjectId": 20606,
    "GroupName": "_API-Folder",
    "ObjectNumber": "L-000002",
    "ObjectName": "Liegenschaft 02",
    "IsDeactivated": false,
    "UsageType": "Nutzungsart",
    "Nrf": 0.0,
```

```
    "Bgf": 0.0,
    "IsVacant": false,
    "CustomField1": "Kundenspezifisches Zusatzfeld 1 in Objektdaten",
    "CustomField2": "Kundenspezifisches Zusatzfeld 2 in Objektdaten",
    "CostCenterNumber": "Nummer der Kostenstelle",
    "CostCenterName": "Name der Kostenstelle",
    "DatevCostCenter": "DATEV Kostenstelle",
    "ObjectForeignKey": "Id aus dem externen Systeme",
    "ObjectAddress": {
      "Zip": "13189",
      "Street": "Arkonastraße 45-49",
      "City": "Berlin",
      "CountryISO3166": null,
      "State": null
    },
    "Notes": "Notizen zu einem Objekt: Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam",
    "Children": [
      {
        "ObjectId": 20607,
        "GroupName": "_API-Folder",
        "ObjectNumber": "G-000001",
        "ObjectName": "Gebäude 01",
        "IsDeactivated": false,
        "UsageType": "Nutzungsart",
        "Nrf": 0.0,
        "Bgf": 0.0,
        "IsVacant": false,
        "CustomField1": "Kundenspezifisches Zusatzfeld 1 in Objektdaten",
        "CustomField2": "Kundenspezifisches Zusatzfeld 2 in Objektdaten",
        "CostCenterNumber": "Nummer der Kostenstelle",
        "CostCenterName": "Name der Kostenstelle",
        "DatevCostCenter": "DATEV Kostenstelle",
        "ObjectForeignKey": "Id aus dem externen Systeme",
        "ObjectAddress": {
          "Zip": "13189",
          "Street": "Arkonastraße 45-49",
          "City": "Berlin",
          "CountryISO3166": null,
          "State": null
        },
        "Notes": "Notizen zu einem Objekt: Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam",
        "Children": null
      }
    ]
  }
]
```


Unauthorized 401:

- Authentication failed

OK 200

- on success with objectList

3.4. Equpiments (Bauteile)

Über diese API können Bauteile / Equipments im Hausmanager zu bestehenden Objekten verknüpft werden.

3.4.1. Equpiment anlegen

Alles, was benötigt wird, sind „AKSKey“ und „ObjectForeignKey“. „AKSKey“ wird verwendet, um das Bauteil aus den Stammdaten des hausmanagers zu finden und und es zum „ObjectForeignKey“ zu verknüpfen, inklusive aller in den Stammdaten hinterlegten wiederkehrenden Maßnahmen.
Sie senden den „ObjectForeignKey“ und den „AKSKey“, und die API erstellt eine neue Instanz des Bauteils / Equipments. Wenn diese Verbindung bereits besteht (ObjectForeignKey und AKSKey sind identisch), passiert nichts, aber es wird ein Erfolg zurückgegeben.
Bauteile haben eine Reihe von Attributen, die über die RestApi zugeordnet werden sollen. Die unten aufgeführten Beispiele bieten einen Überblick:

Request Method: POST

{{baseUrl}}/api/equipment-management/equipments

Beipspiel:

{{baseUrl}}/api/equipment-management/equipments


```
{
 "AksKey": "BC-SST-999",
 "ObjectForeignKey": "L-000001-20605",
 "ExternalId": "Externe ID des Bauteils",
 "InventoryLabel": "TGM",
 "InventoryNumber": "Inventarnummer wenn vorhanden",
 "ScanCode": "Scnacode wenn vorhanden",
 "Manufacturer": "net-haus tech",
 "InventoryName": "TGM",
 "InventoryType": "",
 "SerialNumber": "Seriennummer wenn vorhanden",
 "WarrantyDate": "2024-09-30",
 "Note": "Lorem Ipsum",
 "StartDateRecurringTasks": "2024-04-01"
}
```


3.4.1.1. Eingabeparameter

Name	Datentyp	Länge	Pflicht	Beschreibung
AksKey	string		X	Eindeutiger Schlüssel für das Bauteil (BC-SST-999)
ObjectForeignKey	string		X	Fremdschlüssel, der auf das übergeordnete Objekt verweist (L-000001-20605)

ExternalId	string			Externe ID des Bauteils
InventoryLabel	string			Kennzeichnung des Inventars (TGM)
InventoryNumber	string			Inventarnummer, wenn vorhanden
ScanCode	string			Scan-Code, wenn vorhanden
Manufacturer	string			Hersteller des Geräts (net-haus tech)
InventoryName	string			Name des Inventars (TGM)
InventoryType	string			Typ des Inventars (leer)
SerialNumber	string			Seriennummer des Geräts, wenn vorhanden
WarrantyDate	string			Garantiedatum des Geräts (2024-09-30)
Note	string			Notiz (Lorem Ipsum)
StartDateRecurringTasks	string			Startdatum wiederkehrender Aufgaben (2024-04-01)

3.4.1.2. Ausgabe / http Status Codes

Name	Datentyp	Länge	Bemerkung
id	string	255	die interne Bauteil ID des Hausmanagers

Bad Request 400:
- ForeignKey is required
- AksKey is required
- Object with foreign key not found
- AksKey not found

Unauthorized 401:
- Authentication failed

Not Allowed 403:
- User has no permission for object
- Object is deactivated

Internal Server Error 500
- Server had error during creation of node

Created 201
- returns Id of created node

3.4.2. Equpiment aktualisieren

Das Datenmodell für die Aktualisierung ist ähnlich dem Einfügen. Es hat keinen AKSKey und Object-ForeignKey, sondern stattdessen die Id des Bauteils, welches das einzige obligatorische Feld ist. Alles andere ist optional, was bedeutet, dass der Aufrufer nur die Felder senden muss, die er aktualisieren möchte. Unten sind einige gültige JSONs aufgeführt, die der Aufrufer senden könnte.

Request Method: PATCH

{{baseUrl}}/api/equipment-management/equipments/{id}

Beipspiel:

{{baseUrl}}/api/equipment-management/equipments/87465

```
{
  "id": 87465, //required
```

```
  "externalId": "EX-123",
  "inventoryLabel": "", //Namenszusatz
  "inventoryNumber": "",
  "scanCode": "",
  "manufacturer": "", //Hersteller
  "inventoryName": "", //Bezeichnung
  "inventoryType": "",
  "serialNumber": "",
  "warrantyDate": "", //Gewährleistung bis
  "note": ""
}
```

3.4.2.1. Eingabeparameter

Name	Datentyp	Länge	Pflicht	Beschreibung
Id	Long		X	Id des Bauteils vom Hausmanagers
ExternalId	string			Externe ID des Bauteils
InventoryLabel	string			Kennzeichnung des Inventars (TGM)
InventoryNumber	string			Inventarnummer, wenn vorhanden
ScanCode	string			Scan-Code, wenn vorhanden
Manufacturer	string			Hersteller des Geräts (net-haus tech)
InventoryName	string			Name des Inventars (TGM)
InventoryType	string			Typ des Inventars (leer)
SerialNumber	string			Seriennummer des Geräts, wenn vorhanden
WarrantyDate	string			Garantiedatum des Geräts (2024-09-30)
Note	string			Notiz (Lorem Ipsum)
StartDateRecurringTasks	string			Startdatum wiederkehrender Aufgaben (2024-04-01)

3.4.2.2. Ausgabe / http Status Codes

Bad Request 400:

- Id not found
- Invalid date format

Unauthorized 401:

- Authentication failed

Not Allowed 403:

- User has no permission for object
- Object is deactivated
- Node is deactivated

Internal Server Error 500

- Server had error during update of object

No Content 204

- on success

3.4.3. Equpiment deaktivieren

Wenn ein Bauteil deaktiviert werden soll, muss nurnur die ID oder IDs des/deaktivierten Bauteil gesendet werden. Wenn das Bauteil untergeordnete Bauteile hat, werden auch alle untergeordneten Bauteile deaktiviert.
Alle offenen Aufgaben, die mit diesem Bauteil verbunden sind, werden geschlossen (Status "erledigt"), einschließlich dem Einfügen von Teilerledigungen und Protokolle für die Aufgaben.
Alle Maßnahmen werden mit Protokoll deaktiviert.

Request Method: DELETE

{{baseUrl}}/api/equipment-management/equipments

Beipspiel:

{{baseUrl}}/api/equipment-management/equipments

Body:

```
{
  "Id": 87456
}
```

3.4.3.1. Ausgabe / http Status Codes

Bad Request 400:
- Id not found

Unauthorized 401:
- Authentication failed

Not Allowed 403:
- user has no "permission" for object

Internal Server Error 500
- Server had error during update of object

No Content 204
- on success

3.4.4. Ein Equpiment abfragen

Mit dem Get-Endpunkt erhalten Sie alle Informationen zu einem Bauteil für die gesendete ID.

Request Method: GET

{{baseUrl}}/api/equipment-management/equipments/{equipmentId}

Beipspiel:

{{baseUrl}}/api/equipment-management/equipments/87465

3.4.4.1. Ausgabe / http Status Codes

Bad Request 400:
- Id not found

Unauthorized 401:
- Authentication failed

Not Allowed 403:
- User has no permission for object

Internal Server Error 500
- Server had error during creation of object

Ok 200
- returns list

3.4.5. Eine Liste von Equpiments abfragen

Mit dem GET-Endpunkt können alle Bauteile abgefragt werden. Berücksitigung der Objektfreigabe erfolgt.

Request Method: GET

{{baseUrl}}/api/equipment-management/equipments

Beipspiel:

{{baseUrl}}/api/equipment-management/equipments

3.4.5.1. Ausgabe / http Status Codes

Bad Request 400:

- Id not found

Unauthorized 401:

- Authentication failed

Not Allowed 403:

- User has no permission for object

Internal Server Error 500

- Server had error during creation of object

Ok 200

- returns list

3.5. Wiederkehrende Maßnahmen

Der allgemeine Zweck der API für wiederkehrende Maßnahmen besteht darin, CRUD-Operationen für rhythmusbasierte wiederkehrende Maßnahmen zu ermöglichen, die mit Bauteilen verknüpft sind. Ein Anwendungsfall könnte beispielsweise die regelmäßige Überprüfung einer technischen Analge oder die wöchentliche Reinigung von Objekten sein.

3.5.1. Wiederkehrende Maßnahmen zum Bauteil anlegen

Es sind 3 Attribute Pflicht zum Anlegen von wiederkehrenden Maßnahmen zu existierenden Bauteilen.

- Die Bauteil ID des Hausmanagers
- Der Betreff
- Der Rhythmus

Der Rhythmus entspricht den Angaben des Hausmanagers. Es stehen fünf verschiedene Typen zur Verfügung:

- date(one time events)
- daily
- weekly
- monthly
- yearly

Request Method: POST

{{baseUrl}}/api/equipment-management/recurring-tasks

Beipspiel:

{{baseUrl}}/api/equipment-management/recurring-tasks

```
{
  "equipmentId": 87465, //required
  "category1": "BC-Terminserien",
  "category2": "",
```

```
  "serviceDomain": "100 Objektleitung",
  "subject": "Reinigung Treppenhaus", //required
  "description": "Hier steht Text",
  "externalCostCenter": "12345",
  "rhythm": { //required
     "type": "date",
     "startDate": "2024-04-09"
  }
}
```

3.5.1.1. Eingabeparameter

Name	Datentyp	Länge	Pflicht	Beschreibung
EquipmentId	integer		X	Die ID des Bauteils
Category1	string			Kategorie 1 des Bauteils
Category2	string			Kategorie 2 des Bauteils
ServiceDomain	string			Servicebereich des Bauteils
Subject	string			Betreff der wiederkehrenden Maßnahme
Description	string			Beschreibung der wiederkehrenden Maßnahme
ExternalCostCenter	string			Externe Kostenstelle des Bauteils
Rhythm.Type	string		X	Typ des Rhythmus
Rhythm.StartDate	string		X	Startdatum des Rhythmus
Rhythm.EndDate	string			Enddatum des Rhythmus
Rhythm.Time	string			Zeit des Rhythmus
Rhythm.Interval.Days	integer			Anzahl der Tage im Intervall
Rhythm.Interval.Weeks	integer			Anzahl der Wochen im Intervall
Rhythm.Interval.WeekDays	array of strings			Wochentage im Intervall
Rhythm.Interval.WithMaintenanceDay	boolean			Mit Wartungstag im Intervall?
Rhythm.Interval.IntervalStartDate	string			Startdatum des Intervalls
Rhythm.Interval.IntervalEndDate	string			Enddatum des Intervalls
Rhythm.Interval.Months	integer			Anzahl der Monate im Intervall
Rhythm.Interval.DayOfMonth	integer			Tag des Monats im Intervall
Rhythm.Interval.Years	integer			Anzahl der Jahre im Intervall
Rhythm.Interval.January	integer			Januar im Intervall
Rhythm.Interval.February	integer			Februar im Intervall
Rhythm.Interval.March	integer			März im Intervall
Rhythm.Interval.April	integer			April im Intervall
Rhythm.Interval.May	integer			Mai im Intervall
Rhythm.Interval.June	integer			Juni im Intervall
Rhythm.Interval.July	integer			Juli im Intervall

Rhythm.Interval.August	integer			August im Intervall
Rhythm.Interval.September	integer			September im Intervall
Rhythm.Interval.October	integer			Oktober im Intervall
Rhythm.Interval.November	integer			November im Intervall
Rhythm.Interval.December	integer			Dezember im Intervall

Date:
Für ein einmaliges Ereignis müssen Sie lediglich das Datum mit dem Startdatum angeben. Die Aufgabe wird am angegebenen Datum geplant, in diesem Fall am 16. Januar 2024 um 12:00 Uhr UTC.

```
 "rhythm": {
   "type": "date",
   "startDate": "2024-01-16",
   "time": "12:00:00Z" //optional
}
```

Daily
Dies erstellt eine wiederkehrende Aufgabe, die jeden Tag um 09:00 Uhr UTC beginnend mit dem Startdatum geplant wird. Wenn das Enddatum mit angegeben wird, werden maximal bis zum Enddatum auch Aufgaben generiert (durch eine Funktion im Hausmanager). Das folgende Beispiel würde eine wiederkehrende Aufgabe alle zwei Tage ab dem 16. Januar bis zum 21. Januar erstellen. Das Fehlen eines Intervalls entspricht Tagen: 1 (jeden Tag), und auch das Enddatum ist optional.

```
"rhythm": {
   "type": "daily",
   "startDate": "2024-01-16",
   "endDate": "2024-01-21",
   "time": "08:00:00Z"
   "interval": {
     "days": 2
   }
 }
```

Weekly
Der Typ wöchentlich funktioniert ähnlich wie der tägliche Typ. Das unten genannte JSON würde eine zweiwöchentliche Aufgabe zwischen dem 16. Januar und dem 16. Februar 2024 erstellen. Durch die Definition der Wochentage kann auch entschieden werden, wann die Aufgabe geplant werden soll. Zur genauen Spezifiaktion: alle 2 Wochen am Donnerstag und Freitag. Die Angabe der Wochentage erlaubt es auch mehr als ein Ereignis pro Woche zu haben und eine wiederkehrende Aufgabe wie Montag-Mittwoch-Freitag zu haben, die alle zwei Tage ist, aber das Wochenende auslässt oder Aufgaben nur am Samstag und Sonntag hat. Die möglichen Optionen sind: ["Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday", "Sunday", "MaintenanceDay"]. Die letzte Option ist eine Funktion des Hausmanagers, bei der Sie einen bestimmten Wochentag in jedem Objekt definieren können, an dem wiederkehrende Aufgaben stattfinden sollen.

```
 "rhythm": {
   "type": "weekly",
   "startDate": "2024-01-16",
```

```
  "endDate": "2024-02-16",
  "time": "09:00:00Z",
  "interval": {
     "weeks": 2
   "weekDays": [
        "Tuesday",
        "Friday"
     ],
     "intervalStartDate": "2024-01-16",
     "intervalEndDate": "2024-02-16",
  }
```


Monthly

Wenn der monatliche Typ gewählt wird, haben Sie die Möglichkeit, ein Intervall und den Tag des Monats auszuwählen, an dem die Aufgabe geplant werden soll. In diesem Fall hätten wir eine wiederkehrende Aufgabe am 20. Januar 2024 und dann alle 3 Monate bis zum 16. Januar 2025.


```
"rhythm": {
   "type": "monthly",
   "startDate": "2024-01-16",
   "endDate": "2025-01-16",
   "interval": {
      "months": 3,
      "dayOfMonth": 20
   }
}
```


Wenn kein Intervall angegeben wird, entspricht dies:


```
 "rhythm": {
   "type": "monthly",
   "startDate": "2024-01-16",
   "endDate": "2025-01-16",
   "interval": {
      "months": 1,
      "dayOfMonth": 1
   }
}
```


Yearly

Das oben unten genannte JSON würde eine Aufgabe alle 4 Jahre ab dem 16. Januar 2024 erstellen. Im Hausmanager können jedoch mehr als ein Tag pro Jahr angegeben werden, z. B. am 1. Mai und am 5. Oktober mit einem jährlichen Rhythmus, jedoch mit der Einschränkung, dass pro Monat nur ein Datum angegeben werden kann. Um diesen Umstand zu berücksichtigen, führen wir neue optionale Felder in unser Intervall-Objekt ein:


```
  "rhythm": {
   "type": "yearly",
   "startDate": "2024-01-16",
   "endDate": "2030-01-16",
```

```
  "interval": {
    "years": 4
  }
}
```

3.5.1.2. Ausgabe / http Status Codes

Bad Request 400:
- equipmentId not found
- no subject provided
- invalid rhythm provided
- invalid Kategorie1
- invalid Kategorie2
- invalid Leistungsbereich

Unauthorized 401:
- Authentication failed

Not Allowed 403:
- user has no "permission" for object

Internal Server Error 500
- Server had error during creation of object

Created 201
- returns Id of created task

3.5.2. Wiederkehrende Maßnahmen zum Bauteil aktualisieren

Der Request ist ähnlich dem POST. Die ID der wiederkehrenden Maßnahme ist obligatorisch. Es müssen auch nur die Felder gesendet werden, die geändert werden sollen. Über PATCH kann die wiederkehrende Maßnahme deaktiviert werden. Das bedeutet, dass auf Basis dieser wiederkehrenden Maßnahme keine neuen Aufgaben mehr erzeugt werden. Gleiches gilt für das Aktivieren. Wenn versucht wird eine deaktivierte wiederkehrende Maßnahme zu aktualisieren wird eine Fehlermeldung als Antwort ausgegeben.

Request Method: PATCH

{{baseUrl}}/api/equipment-management/recurring-tasks/{id}

Beipspiel:

{{baseUrl}}/api/equipment-management/recurring-tasks/120962

```
{
  "id": 120962, // required
  "category1": "Kategerorie1",
  "category2": "Kategorie2",
  "serviceDomain": "Leistungsbereich",
  "subject": "Reinigung Treppenhaus",
  "description": "Hier steht Text",
  "externalCostCenter": "12345",
  "isDeactivated": false,
  "rhythm": {
    "type": "yearly",
    "startDate": "2024-01-16",
```

```
    "endDate": "2030-01-16",
    "interval": {
      "years": 4,
      "may": 1,
      "october": 5
    }
  }
}
```


3.5.2.1. Ausgabe / http Status Codes

Bad Request 400:
- id not found
- invalid rhythm provided
- invalid Kategorie1
- invalid Kategorie2
- invalid Leistungsbereich

Unauthorized 401:
- Authentication failed

Not Allowed 403:
- user has no permission for object

Internal Server Error 500
- Server had error during update of task

No content 204
- (on sucess)

3.5.3. Eine Wiederkehrende Maßnahmen zum Bauteil abfragen

Request Method: GET

{{baseUrl}}/api/equipment-management/recurring-tasks/{id}

Beipspiel:

{{baseUrl}}/api/equipment-management/recurring-tasks/120962

3.5.3.1. Eingabeparameter

3.5.3.2. Ausgabe / http Status Codes

Bad Request 400:
- equipmentId not found
- objectId not found

Unauthorized 401:
- Authentication failed

Not Allowed 403:
- user has no "permission" for object/component

Internal Server Error 500
- Server had error during creation of object

OK 200
- returns Object

[

```
  {
    "Id": 120962,
    "EquipmentId": 87465,
    "ObjectId": 20605,
    "Category1": "BC-Terminserien",
    "Category2": "",
    "ServiceDomain": "100 Objektleitung",
    "Subject": "TGM",
    "Description": "",
    "ExternalCostCenter": null,
    "IsDeactivated": "False",
    "Rhythm": {
      "Type": "monthly",
      "StartDate": "2024-04-01",
      "EndDate": "9999-12-31",
      "Time": "00:00z",
      "Interval": {
        "Days": 0,
        "Weeks": 0,
        "WeekDays": null,
        "WithMaintenanceDay": false,
        "IntervalStartDate": null,
        "IntervalEndDate": null,
        "Months": 1,
        "DayOfMonth": 0,
        "Years": 0,
        "January": 0,
        "February": 0,
        "March": 0,
        "April": 0,
        "May": 0,
        "June": 0,
        "July": 0,
        "August": 0,
        "September": 0,
        "October": 0,
        "November": 0,
        "December": 0
      }
    }
  }
]
```


3.5.4. Eine Liste von Wiederkehrende Maßnahmen zum Bauteil abfragen

Dieser Endpunkt liefert eine Liste von wiederkehrenden Maßnahmen zurück und berücksichtigt die Objektfreigabe. Query Parameter sind

- equipmentId
- objectId
- withchildren

Wenn equipmentId benutzt wird, bedeutet es: alle Maßnahmen zu einem Bauteil. Wenn objectId benutzt wird, bedeutet es: alle Maßnahmen zu allen Bauteilen des Objektes. Wenn withChilderen benutzt wird, bedeutet es: dass auch alle untergeordneten. Die Verwendung von equipmentId UND objectId in einem Request ist nicht erlaubt.

Request Method: GET

{{baseUrl}}/api/equipment-management/recurring-tasks

Beipspiel:

{{baseUrl}}/api/equipment-management/recurring-tasks?equipmentId=87465&withChildren=1

{{baseUrl}}/api/equipment-management/recurring-tasks?objectId=20605&withChildren=1


```
[
  {
    "Id": 120962,
    "EquipmentId": 87465,
    "ObjectId": 20605,
    "Category1": "BC-Terminserien",
    "Category2": "",
    "ServiceDomain": "100 Objektleitung",
    "Subject": "TGM",
    "Description": "",
    "ExternalCostCenter": null,
    "IsDeactivated": "False",
    "Rhythm": {
      "Type": "monthly",
      "StartDate": "2024-04-01",
      "EndDate": "9999-12-31",
      "Time": "00:00z",
      "Interval": {
        "Days": 0,
        "Weeks": 0,
        "WeekDays": null,
        "WithMaintenanceDay": false,
        "IntervalStartDate": null,
        "IntervalEndDate": null,
        "Months": 1,
        "DayOfMonth": 0,
        "Years": 0,
        "January": 0,
        "February": 0,
        "March": 0,
        "April": 0,
        "May": 0,
        "June": 0,
        "July": 0,
        "August": 0,
        "September": 0,
        "October": 0,
        "November": 0,
```

```
            "December": 0
          }
        }
      },
      {
        "Id": 120963,
        "EquipmentId": 87465,
        "ObjectId": 20605,
        "Category1": "BC-Terminserien",
        "Category2": "",
        "ServiceDomain": "100 Objektleitung",
        "Subject": "Reinigung Treppenhaus",
        "Description": "Hier steht Text",
        "ExternalCostCenter": "12345",
        "IsDeactivated": "False",
        "Rhythm": {
          "Type": "date",
          "StartDate": "2024-04-09",
          "EndDate": "9999-12-31",
          "Time": null,
          "Interval": {
            "Days": 0,
            "Weeks": 0,
            "WeekDays": null,
            "WithMaintenanceDay": false,
            "IntervalStartDate": null,
            "IntervalEndDate": null,
            "Months": 0,
            "DayOfMonth": 0,
            "Years": 0,
            "January": 0,
            "February": 0,
            "March": 0,
            "April": 0,
            "May": 0,
            "June": 0,
            "July": 0,
            "August": 0,
            "September": 0,
            "October": 0,
            "November": 0,
            "December": 0
          }
        }
      }
]
```

3.6. Material-Kostenstellen

Über diese API können Kostenstellen in Bezug auf Artikel / Material hinzugefügt werden.

3.6.1. Material-Kostenstelle anlegen

Request Method: POST

{{baseUrl}}/api/material-management/cost-centers

Beipspiel:

{{baseUrl}}/api/material-management/cost-centers

3.6.1.1. Eingabeparameter

Name	Datentyp	Länge	Pflicht	Beschreibung
Key	string		X	Kennung
Text	string			Beschreibung
Inventory	string			Lager

3.6.1.2. Ausgabe / http Status Codes

Name	Datentyp	Länge	Pflicht	Beschreibung
Id	string			Id der erstellten Material-Kostenstellen

Bad Request 400:

- no key provided
- no text provided
- key already exists

Unauthorized 401:

- Authentication failed

Not Allowed 403:

- user has no "permission" for object.

Internal Server Error 500

- Server had error during creation of object

Created 201

- returns Id of created cost centers, or list of created cost center ids.
  (if is creating a list, if something is not valid, do not create anything).

3.6.2. Material-Kostenstelle aktualisieren

Request Method: POST

{{baseUrl}}/api/material-management/cost-centers

Beipspiel:

{{baseUrl}}/api/material-management/cost-centers


```
{
 "Id": 1,
 "Key": "aktualisierter Key",
 "Text": "aktualiserter Text",
 "Inventory": "aktualisiertes Lager"
}
```

3.6.2.1. Eingabeparameter

Name	Datentyp	Länge	Pflicht	Beschreibung
Id	Integer		X	Id der Material-Kostenstellen
Key	string			Kennung
Text	string			Beschreibung
Inventory	string			Lager

3.6.2.2. Ausgabe / http Status Codes

Bad Request 400:

- id not found
- id not provided
- provided key already in use
- text is empty, value is required

Unauthorized 401:

- Authentication failed

Not Allowed 403:

- user has no permission for object

Internal Server Error 500

- Server had error during update of cost centers

No content 204

- (on sucess)

3.6.3. Eine Material-Kostenstelle abfragen

Request Method: GET

{{baseUrl}}/api/material-management/cost-centers/{id}

Beipspiel:

{{baseUrl}}/api/material-management/cost-centers/1

Query-Parameter:

- Key
- Text

Beipspiel:

{{baseUrl}}/api/material-management/cost-centers?key=aktualisierter Key
{{baseUrl}}/api/material-management/cost-centers?text=aktualisierter text

3.6.3.1. Ausgabe / http Status Codes


```
{
    "Id": 1,
    "Key": "aktualisierter Key",
    "Text": "aktualiserter Text",
    "Inventory": "aktualisiertes Lager"
}
```


Bad Request 400:

- id not found

Unauthorized 401:

- Authentication failed

Not Allowed 403:

- user has no "permission" for object/component

Internal Server Error 500
- Server had error during creation of object

OK 200
- returns Object

3.7. Material-Produktgruppen

Über diese API können Material-Produktgruppen erstellt, aktualisiert und abgefragt werden.

3.7.1. Material-Produktgruppe anlegen

Produktgruppen können einem Mandanten zugewiesen sein, müssen es aber nicht. Zur Zuordnung ist die clientId oder der ClientName notwendig. Einer der beiden muss gesetzt sein. Beides wird vom Projektpartner mitgeteilt.

Request Method: POST

{{baseUrl}}/api/material-management/groups

Beipspiel:

{{baseUrl}}/api/material-management/groups

```
{
 "Key": "Kennung",
 "Name": "Bezeichnung",
 "IsSalary": true, //Lohn?
 "ClientId": 1,
 "ClientName": "gepe Gebäudedienste Peterhoff Gmbh"
}
```

3.7.1.1. Eingabeparameter

Name	Datentyp	Länge	Pflicht	Beschreibung
Key	string	-	X	Eindeutiger und erforderlicher Schlüsselwert für das Objekt.
Name	string	-	X	Erforderlicher Name des Objekts.
IsSalary	boolean	-	Nein	Gibt an, ob es sich um einen Gehaltsdatensatz handelt. Default ist immer false
ClientId	integer	-	Ja	Erforderliche ID des Clients.
Client-Name	string	-	Nein	Name des Clients (optional).

3.7.1.2. Ausgabe / http Status Codes

HTTP-Statuscode	Beschreibung
400	Bad Request
	- Kein Schlüsselwert angegeben

	- Kein Name angegeben
	- Schlüsselwert existiert bereits
	- Keine tenantId angegeben
	- Angegebene clientId existiert nicht
	- Angegebener clientName existiert nicht
	- Weder clientId noch clientName angegeben
401	Unauthorized
	- Authentifizierung fehlgeschlagen
403	Not Allowed
	- Benutzer hat keine Berechtigung für das Objekt
500	Internal Server Error
	- Serverfehler während der Erstellung des Objekts
201	Created
	- Rückgabe der ID(s) der erstellten Materialgruppen oder Liste der erstellten Materialgruppen-IDs. (Wenn eine Liste erstellt wird und etwas ungültig ist, wird nichts erstellt.)

3.7.2. Material-Produktgruppe aktualisieren

Request Method: PATCH

{{baseUrl}}/api/material-management/groups

Beipspiel:

{{baseUrl}}/api/material-management/groups

```
{
  "Id": 0,
  "Key": "string",
  "Name": "string",
  "IsSalary": true,
  "ClientId": 0,
  "ClientName": "string"
}
```

3.7.2.1. Eingabeparameter

Name	Datentyp	Länge	Pflicht	Beschreibung
Id	String		X	Id der Produktgruppe
Key	string	-		Eindeutiger und erforderlicher Schlüsselwert für das Objekt.
Name	string	-		Erforderlicher Name des Objekts.
IsSalary	boolean	-	Nein	Gibt an, ob es sich um einen Gehaltsdatensatz handelt. Default ist immer false

ClientId	integer	-	Ja	Erforderliche ID des Clients.
Client-Name	string	-	Nein	Name des Clients (optional).

3.7.2.2. Ausgabe / http Status Codes

HTTP-Statuscode	Beschreibung
400	Bad Request
	- ID nicht gefunden
	- ID nicht angegeben
	- Schlüssel ist leer, Wert erforderlich
	- Angegebene clientId nicht gefunden
	- Angegebener clientName nicht gefunden
401	Unauthorized
	- Authentifizierung fehlgeschlagen
403	Not Allowed
	- Benutzer hat keine Berechtigung für das Objekt
500	Internal Server Error
	- Serverfehler während der Aktualisierung von Material-gruppen
204	No Content
	- (Bei Erfolg)

3.7.3. Eine Material-Produktgruppe abfragen

Request Method: GET

{{baseUrl}}/api/material-management/groups/{id}

Beipspiel:

{{baseUrl}}/api/material-management/groups/1

Query-Parameter

- Key
- Name
- clientId

3.7.3.1. Ausgabe / http Status Codes

HTTP-Statuscode	Beschreibung
400	Bad Request
	- ID nicht gefunden
401	Unauthorized

	- Authentifizierung fehlgeschlagen
403	Not Allowed
	- Benutzer hat keine Berechtigung für das Objekt/Komponente
500	Internal Server Error
	- Serverfehler während der Erstellung des Objekts/Komponente
200	OK
	- Gibt das DTO zurück

3.8. Material / Artikel

Über diese API können Artikel im Hausmanager angelegt, aktualisiert und abgefragt werden.

3.8.1. Material / Artikel anlegen

Über diese API können einzelne oder mehrere Artikel angelegt werden.

Request Method: POST

{{baseUrl}}/api/material-management/material

Beipspiel:

{{baseUrl}}/api/material-management/material

```
{
  "Key": "01.01.0310",
  "ExternalIdentifier": "externe Kennung",
  "ShortText": "DGUV V3 Prüfung je Gerät inkl.Barcode",
  "LongText": "DGUV V3 Prüfung je Gerät inkl.Barcode / Lorem Ipsum",
  "Unit": "Stk",
  "SalesTax": "Standard",
  "Manufacturer": "string",
  "PurchasePrice": 10,
  "Price": 20,
  "ProductGroup": [
  ],
  "RestrictedForProperties": [
  ],
  "ClientName": null
}
```

3.8.1.1. Eingabeparameter

Name	Datentyp	Länge	Pflicht	Beschreibung
Key	string	-	X	Kennung des Artikels im Hausmanager / Kann auch mit der Kennung des externen Systems befüllt werden wenn unique
ExternalIdentifier	string	-		externe Kennung

ShortText	string	-	X	Kurztext - DGUV V3 Prüfung je Gerät inkl.Barcode
LongText	string	-		Langtext - DGUV V3 Prüfung je Gerät inkl.Barcode / Lorem Ipsum
Unit	string	-		Stk
SalesTax	string	-		Standard (siehe API Umsatzsteuerschlüssel)
Manufacturer	string	-		Hersteller
PurchasePrice	number	-		EK
Price	number	-		VK
ProductGroup	array	-		Produktgruppe (Siehe API Produktgruppen)
RestrictedForProperties	array	-		Angabe von Ids der Objekte im Hausmanager (Siehe Objekte anlegen). Mit dieser Angabe können Artikel auf einzelne Objekte freigegeben werden. Damit sind nur diese in den Objekten auswählbar im Hausmanager
ClientName	string	-		Mandant ((wird vom Projektpartner mitgeteilt)

3.8.1.2. Ausgabe / http Status Codes

HTTP-Statuscode	**Beschreibung**
400	Bad Request
	- Kein externalIdentifier angegeben
	- Kein shortText angegeben
	- salesTax nicht gefunden. Mögliche Werte: {Liste der möglichen Umsatzsteuersätze}
	- productGroup nicht gefunden. Mögliche Werte: {Liste der möglichen Produktgruppen}
	- externalIdentifier existiert bereits (nachfragen)
	- clientName nicht gefunden
401	Unauthorized
	- Authentifizierung fehlgeschlagen
403	Not Allowed
	- Benutzer hat keine Berechtigung für das Objekt
500	Internal Server Error
	- Serverfehler während des Abrufs des Objekts
201	Created
	- Gibt die ID des erstellten Materials zurück oder eine Liste der erstellten Material-IDs.
	(Wenn eine Liste erstellt wird und etwas ungültig ist, wird nichts erstellt.)

3.8.2. Material / Artikel aktualisieren

Request Method: PATCH

{{baseUrl}}/api/material-management/material

Beipspiel:

{{baseUrl}}/api/material-management/material


```
{
  "MaterialId": 4711,
  "Key": "01.01.0310",
  "ExternalIdentifier": "externe Kennung",
  "ShortText": "DGUV V3 Prüfung je Gerät inkl.Barcode",
  "LongText": "DGUV V3 Prüfung je Gerät inkl.Barcode / Lorem Ipsum",
  "Unit": "Stk",
  "SalesTax": "Standard",
  "Manufacturer": "string",
  "PurchasePrice": 10,
  "Price": 20,
  "ProductGroup": [
  ],
  "RestrictedForProperties": [
  ],
  "ClientName": null
}
```


3.8.2.1. Eingabeparameter

Name	Datentyp	Länge	Pflicht	Beschreibung
MaterialId	number		X	ID des Materials aus dem Hausmanager
Key	string			Kennung des Artikels im Hausmanager / Kann auch mit der Kennung des externen Systems befüllt werden wenn unique
ExternalIdentifier	string			externe Kennung
ShortText	string			Kurztext - DGUV V3 Prüfung je Gerät inkl.Barcode
LongText	string			Langtext - DGUV V3 Prüfung je Gerät inkl.Barcode / Lorem Ipsum
Unit	string			Stk
SalesTax	string			Standard (siehe API Umsatzsteuerschlüssel)
Manufacturer	string			Hersteller
PurchasePrice	number			EK
Price	number			VK
RestrictedForProperties	array			Produktgruppe (Siehe API Produktgruppen)

ProductGroups	array			Angabe von Ids der Objekte im Hausmanager (Siehe Objekte anlegen). Mit dieser Angabe können Artikel auf einzelne Objekte freigegeben werden. Damit sind nur diese in den Objekten auswählbar im Hausmanager
ClientName	string			Mandant ((wird vom Projektpartner mitgeteilt)

3.8.2.2. Ausgabe / http Status Codes

HTTP-Statuscode	Beschreibung
400	Bad Request
	- id nicht gefunden
	- id nicht angegeben
401	Unauthorized
	- Authentifizierung fehlgeschlagen
403	Not Allowed
	- Benutzer hat keine Berechtigung für das Objekt
500	Internal Server Error
	- Serverfehler während des Updates der Aufgabe
204	No Content
	- Erfolgreiche Anfrage, keine Inhalte zurückgegeben

3.8.3. Ein Material / Artikel abfragen

Request Method: GET

{{baseUrl}}/api/material-management/material/{materialId}

Beipspiel:

{{baseUrl}}/api/material-management/material/189


```
[
  {
    "MaterialId": 189,
    "Key": "04.1.0010",
    "ExternalIdentifier": "",
    "ShortText": "Langtext des Artikels",
    "LongText": "",
    "Unit": "h",
    "SalesTax": null,
    "Manufacturer": "",
    "PurchasePrice": 0.0,
    "Price": 55.0,
    "RestrictedForProperties": [],
    "ProductGroups": [],
    "ClientName": null
  }
]
```

3.8.3.1. Ausgabe / http Status Codes

HTTP-Statuscode	Beschreibung
400	Bad Request
	- ID nicht gefunden
401	Unauthorized
	- Authentifizierung fehlgeschlagen
403	Not Allowed
	- Benutzer hat keine Berechtigung für Objekt/Komponente
500	Internal Server Error
	- Serverfehler während der Erstellung des Objekts
200	OK
	- Gibt das Datenübertragungsobjekt (DTO) zurück

3.8.4. Eine Liste von Material / Artikel abfragen

Request Method: GET

{{baseUrl}}/api/material-management/material

Beipspiel:

{{baseUrl}}/api/material-management/material

Query Parameter

- externalIdentifier
- manufacturer
- key

{{baseUrl}}/api/material-management/material?manufacturer=Hersteller
{{baseUrl}}/api/material-management/material?key=04.1.0010


```
[
  {
    "MaterialId": 189,
    "Key": "04.1.0010",
    "ExternalIdentifier": "",
    "ShortText": "Langtext des Artikels",
    "LongText": "",
    "Unit": "h",
    "SalesTax": null,
    "Manufacturer": "",
    "PurchasePrice": 0.0,
    "Price": 55.0,
    "RestrictedForProperties": [],
    "ProductGroups": [],
    "ClientName": null
  }
]
```

3.9. Umsatzsteuerschlüssel

3.9.1. Umsatzsteuerschlüssel abfragen

Mit dieser API können die Umsatzsteuerschlüssel auf dem Hausmanager abgefragt werden.

Request Method: GET

{{baseUrl}}/api/sale-tax-management/sale-taxes

Beipspiel-Response:


```
[
  {
    "TaxId": 1,
    "TaxRate": 19.00,
    "Label": "Standard",
    "IsDefault": true,
    "RevenueAccount": null
  }
]
```


3.10. Transferdaten

Der allgemeine Zweck besteht darin, Benutzern das Abrufen und Aktualisieren von TemporaryTransferData-Objekten zu ermöglichen. Das sind Daten die vom Hausmanager speziell für ein externes System bereitgestellt und abgefragt werden können. Das Prinzip erfordert es, dass nach der erfolgreichen Verarbeitung eines solchen Datensatzes im externen System der Transferdatensatz über PATCH aktualisiert wird, damit dieser bei einer erneuten Abfrage nicht noch einmal abgeholt wird.

3.10.1. Transferdaten abfragen

Ermöglicht es dem User, Informationen für eine bestimmte Zeile oder alle Zeilen (die dem Filter und der Autorisierung entsprechen) abzurufen. Es werden nur Zeilen mit dem Status "Geplant" zurückgegeben. Im Rückgabe Objekt ist die Payload enthalten. Diese Payload enthält die eigentlichen Nutzdaten und die relevanten Informationen für die Prozesse. Die Payload wird Prozessabhängig aufgebaut. Inhalte werden in Abstimmung mit dem Projektpartner definiert. Eine Beispielausprägung ist unten aufgeführt.
Query parameters:

- id
- interfaceName
- processName

Request Method: GET

{{baseUrl}}/api/transfer-data-management/transfer-data

Beipspiel:

{{baseUrl}}/api/transfer-data-management/transfer-data
{{baseUrl}}/api/transfer-data-management/transfer-data/23
{{baseUrl}}/api/transfer-data-management/transfer-data?interface=Sup3er
{{baseUrl}}/api/transfer-data-management/transfer-data?interface=Sup3er&processName=recurringStuffEvent

3.10.1.1. Ausgabe / http Status Codes


```
[
  {
    "id": 23,
    "interface": "Sup3r",
    "payload": "[{\"ClientId": "3EEDF269-90AF-4356-B0A9-F64DF160DAE0\"}]",
    "date": "2024-01-16T09:21:00Z",
    "processName": "recurringStuffEvent"
  },
  {
    "id": 24,
    "interface": "Sup3r",
    "payload": "[{\"AdHocGuy": "Michael\"}]",
    "date": "2024-01-16T11:22:00Z",
    "processName": "adhocStuffEvent"
  }
]
```


Payload:


```
{
  "payload": [
    {
      "TaskId": 1671245,
      "TaskSubject": "Test",
      "TaskText": "",
      "TaskCategory": "Bedarfsleistung",
      "TaskCategory2": "keine",
      "TaskCraft": "Keins",
      "TaskPlanStart": "2024-02-15T00:00:00",
      "TaskDoneDate": "2024-02-15T00:00:00",
      "TaskCreatedDate": "2024-02-15T16:30:36.933",
      "TaskHighPriority": 0,
      "TaskDuration": 3.00,
      "TaskDurationTarget": 0.00,
      "taskCostCenter": "",
      "BillingType": "singleBilling",
      "MaterialPriceBeforeTax": 12079.95,
      "MaterialPriceAfterTax": 14375.14,
      "TaskWorkTime": 3.00,
      "HomeId": 16417,
      "HomeExternalKey": "TRANSFER-KEY-L",
      "CostUnitDatev": "",
      "HomeName": "Liegenschaft",
      "material": [
        {
          "TaskId": 1671245,
          "Key": "5000",
          "Label": "Gala",
          "Quantity": 14.00,
          "Price": 0.00,
```

```
            "PriceAfterTax": 0.00,
            "Tax": 29.00
          },
          {
            "TaskId": 1671245,
            "Key": "B1",
            "Label": "Wasserkocher",
            "Description": "Wasserkocher Mark IV",
            "Quantity": 145.00,
            "Unit": "Stk",
            "Price": 83.31,
            "PriceAfterTax": 99.14,
            "Tax": 19.00
          }
        ]
      }
  ]
}
```


3.10.2. Transferdaten(-satz) bestätigen

Über diese API können einzelne oder mehrere Transferdatensätze quittiert werden.

Request Method: PATCH

{{baseUrl}}/api/transfer-data-management/transfer-data

Beipspiel:

{{baseUrl}}/api/transfer-data-management/transfer-data


```
{
  "confirmedIds": [
    21,
    41,
    56,
    78
  ]
}
```


3.10.2.1. Ausgabe / http Status Codes

HTTP-Statuscode	Beschreibung
400	Bad Request
	- ID nicht gefunden
	- ID nicht angegeben
401	Unauthorized
	- Authentifizierung fehlgeschlagen
500	Internal Server Error
	- Serverfehler während des Updates von temporaryTransferData

204	No Content
	- Erfolgreiche Anfrage, keine Inhalte zurückgegeben

3.11. Verknüpfung von Adressen mit Objekten (AddressObjectLink)

Mit dieser API können zu bestehenden Objekten des Hausmanagers Adressen verknüpft werden. Das sind u.a. Adressen vom Typ Eigentümer, Rechnung, Nutzer/Bewohner oder die im Hausmanager hinterlegten Leistungsbereiche. Vom Hausmanager vorgegebene gültige Werte sind:

- Owner -> Eigentümer
- Object -> Objektadresse
- Renter -> Mieter (Nutzer/Bewohner)
- Billing -> Rechnung

3.11.1. (AddressObjectLink) anlegen

Request Method: POST

{{baseUrl}}/api/address-object-link-management/address-object-links

Beipspiel:

{{baseUrl}}/api/address-object-link-management/address-object-links

Body:

```
{
    "addressRole": "Elektriker",
    "addressId": 319,
    "objectId": 1313
}
```

3.11.1.1. Eingabeparameter

Name	Datentyp	Länge	Pflicht	Beschreibung
addressRole	number		X	Klartext des Leistungsbereiches – Wird vom Projektpartner mitgeteilt, wenn es sich um individuelle handelt. Die systemweiten sind oben dargestellt
addressId	string		X	ID der Adresse des Hausmanagers
objectId	string		X	ID des Objektes im Hausmanager

3.11.1.2. Ausgabe / http Status Codes

HTTP-Statuscode	Beschreibung
400	Bad Request
401	Unauthorized
	- Authentifizierung fehlgeschlagen
403	Not Allowed
	- Benutzer hat keine Berechtigung für das Objekt
500	Internal Server Error
	- Serverfehler während des Abrufs des Objekts
201	Created
	-id des „AddressObjectLink“

3.11.2. (AddressObjectLink) abfragen

Für das „GET“ gibt es mehrere Varianten:

- Abfrage eines AddressObjectLink
- Abfrage aller AddressAddressObjectLink aus Sicht der Adresse (wo ist diese Adresse verknüpft)
- Abfrage aller ObjectAddressObjectLink aus Sicht des Objektes (welche Adressen sind zum Objekt verknüpft)

Request Method: GET AddressObjectLink

{{baseUrl}}/api/address-object-link-management/address-object-links/{id}

Beipspiel:

{{baseUrl}}/api/address-object-link-management/address-object-links/

Name	Datentyp	Länge	Pflicht	Beschreibung
id	long		X	id des „AddressObjectLink“


```
{
    "Id": 91542,
    "ObjectId": 20605,
    "AddressId": 10042,
    "Role": "Object"
}
```

Request Method: GET AddressAddressObjectLink

{{baseUrl}}/api/address-management/addresses/{addressId}/address-object-links

Beipspiel:

{{baseUrl}}/api/address-management/addresses/10042/address-object-links

Name	Datentyp	Länge	Pflicht	Beschreibung
addressId	long		X	id der Adresse des Hausmanager


```
[
    {
        "Id": 91542,
        "ObjectId": 20605,
        "AddressId": 10042,
        "Role": "Object"
    }
]
```


Request Method: GET ObjectAddressObjectLink

{{baseUrl}} /api/object-management/objects/{objectId}/address-object-links

Beipspiel:

{{baseUrl}}/api/object-management/objects/20605/address-object-links

Name	Datentyp	Länge	Pflicht	Beschreibung
objecId	long		X	id des Objektes aus dem Hausmanager


```
[
    {
        "Id": 91542,
        "ObjectId": 20605,
        "AddressId": 10042,
        "Role": "Object"
    }
]
```


3.11.3. (AddressObjectLink) löschen

3.11.3.1. Eingabeparameter

Name	Datentyp	Länge	Pflicht	Beschreibung
id	long		X	id des „AddressObjectLink“

Request Method: DELETE

{{baseUrl}}/api/address-object-link-management/address-object-links/{id}

Beipspiel:

{{baseUrl}}/api/address-object-link-management/address-object-links/

4. Vorgehensmodell zur Nutzung der APIs

Das folgende Modell geht von einer grundlegenden Neueinrichtung des Hausmanagers über die APIs aus. Zentrale Elemente im Hausmanager sind immer die Objekte und die verknüpften Bauteile. Weitere Besonderheit ist die Möglichkeit Artikel mit Objekten zu verknüpfen, sodass prozessual im Hausmanager nur noch die Artikel in einer Aufgabe zu einem Objekt ausgewählt werden können. Die weiteren Daten und auch die API setzen darauf auf.

Empfehlung für „POST“ (Reihenfolge ist absteigend aufgelistet):

- Anlegen von Material-Kostenstellen (MaterialCostCenter)
- Anlegen von Material-Produktgruppen (MaterialGroup)
- Anlegen von Objekten (Object)
- Anlegen von Material (Material)
- Anlegen von Adressen (Address)
- Verknüpfung von Objekten zu Adressen (AddressObjectLink)
- Anlegen von Bauteilen zu Objekten (Equipments)
- Anlegen von wiederkehrenden Maßnahmen (RecurringTasks)