Energinet
Tonne Kjærsvej 65
DK-7000 Fredericia

+45 70 10 22 44
info@energinet.dk
VAT no. 39 31 50 41

Date:
January 9, 2020

Author:
SSG/JLI

CUSTOMER AND THIRD PARTY API FOR DATAHUB (ELOVERBLIK) – TECHNICAL DESCRIPTION

Document history

Version	Date	Description	Author(s)
1.0	01-05-2019	Initial version	Søren Søndergaard Janine Lindberg
2.0	22-11-2019	The following major changes apply: • Incorrect path for *Get authorizations* (section 2.6.2) has been corrected. • Information about main response structures have been added for *Get time series* (sections 1.6.8 and 2.6.6). • The mentioning of limits on the number of metering points have been removed from the description of all methods supporting multiple metering points in the request, since this has not been implemented. The same applies to the limit on result values for *Get time series* (sections 1.6.8 and 2.6.6). • In the response examples for *Get meter readings* in appendix B, meterNumber has been removed from the top-level structure and inserted into the Readings structure instead. • The JSON response example for *Get time series* in Appendix B has been corrected to specify that the Period structure can be repeated.	Janine Lindberg
3.0	11-12-2019	API location for pre-production environment has been changed, sections 1.2 and 2.2.	Janine Lindberg
4.0	27-12-2019	Spelling mistake in third party API URL for production environment has been corrected, section 2.2.	Janine Lindberg
5.0	09-01-2020	Incorrect paths for several methods have been corrected (sections 1.6.2, 1.6.3, 1.6.4, 1.6.5, 1.6.6, 1.6.7 and 2.6.3).	Janine Lindberg

Table of contents

1. Customer API

1.1 Introduction

This API is intended to be used by customers (i.e. electricity consumers and producers) who want to access their own data from DataHub. To access the data the user must be authorized by use of a token (see section 1.5 for further information). When using the token all endpoints described in this chapter are accessible.

The following data can be requested:

1. List of metering points associated with the user (either linked or not linked), including selected metering point details (master data) (see section 1.6.2)
2. Extended list of details (master data) per metering point (see section 1.6.6)
3. Charge data per metering point (see section 1.6.7)
4. Time series per metering point (see section 1.6.8)
5. Meter readings per metering point (see section 1.6.9)

To request data mentioned in items 2-5 above, the metering points in question must first be actively linked to the user. There are two ways of creating links/relations to metering points:

1. Submit a list of metering points which are registered in DataHub to the user's CPR or CVR number retrievable from the supplied token (see section 1.6.3)
2. Submit a valid combination of metering point id and web access code (see section 1.6.4)

1.2 API location

Environment	URL
Pre-production environment	https://apipreprod.eloverblik.dk/CustomerApi/
Production environment	https://api.eloverblik.dk/CustomerApi/

1.3 Correlation id

It is possible to set a correlation id in the request header using the 'X-User-Correlation-ID' (with an UUID). When provided this id will follow the request and finally end in the response header. This id can be used for tracking the request. In parallel to this id there is another internal id (also a UUID), which is also returned in the response header as 'X-Correlation-ID'.

Please note that the *Get time series* response is subject to a different market message standard. Therefore, only one id (mRID) is returned in this response. This id is an internal id similar to the X-Correlation-ID.

1.4 Error codes and HTTP responses

A list of relevant error codes and HTTP status codes can be found in appendix A. Please note that the codes are subject to change.

1.5 Tokens

Authentication and authorization are handled by using OAuth 2.0 tokens. To get started a refresh token is required. A refresh token for customer API access can be created in the Eloverblik web portal after logging in with NemID as a private or business customer. The token is a long text string (JWT token), which must be copied and stored for use with the system that needs to access the API.

When a refresh token is obtained, the token endpoint can be accessed to create a short-lived data access token (valid for 60 minutes). See section 1.6.1 for further information. For all data access the data access token needs to be supplied in the HTTP header:

Authorization: Bearer <data-access-token>

1.6 Endpoints

1.6.1 Get data access token

Path: /api/Token
Parameters: None
HTTP verb: GET
Response: See example in appendix B (ii. Get data access token / Add relation based on CPR/CVR / Add relation with web access code / Delete relation – String)

Note: Include the refresh token in the HTTP header as mentioned in section 1.5.

1.6.2 Get metering points

This request is used for getting a list of metering points associated with a specific user (either private or business user). If the parameter includeAll is false (default), only metering points actively linked/related to the user (i.e. metering points with existing relations) are returned. If includeAll is true, the list of actively linked/related metering points will be merged with additional non-linked metering points registered in DataHub to the CPR or CVR number of the user. The CPR (private customers) or CVR (business customers) is retrieved by use of the supplied token.

Path: /api/MeteringPoints/MeteringPoints?includeAll={value}
Parameters:

Name	Type	Value
includeAll	Boolean	true | false

HTTP verb: GET
Response: See example in appendix B (iii. Get metering points – MeteringPointPublicApi)

1.6.3 Add relation based on CPR/CVR

This request is used for linking one or more metering points to a user when the metering points are registered to the user's CPR or CVR in DataHub. The system will retrieve the CVR or CPR number related to the supplied token and verify if the supplied metering points are registered in DataHub to the retrieved CPR/CVR. If so, the relations are created. If not, the request is rejected.

The logical flow will be as follows:

1. First use the Get metering points endpoint (as described in section 1.6.2) to get a list of linked and non-linked metering points (i.e metering points with and without relations)
2. Then use the Add relation endpoint to link (i.e. create relations for) all or some of the non-linked metering points

Path: /api/MeteringPoints/MeteringPoint/Relation/Add
Parameters:

Name	Type	Value
meteringPointIds	String array	List of metering point ids See example in appendix B (i. List of metering points)

HTTP verb: POST
Response: String with the value of the metering point id. See example in appendix B (ii. Get data access token / Add relation based on CPR/CVR / Add relation with web access code / Delete relation – String).

1.6.4 Add relation with web access code

This request is used for linking a metering point to a user by use of the web access code (WAC) for the metering point. The WAC is provided by the electricity supplier and can typically be found on the electricity bill. Adding relations by use of WAC is relevant for many metering points associated with private users/customers since many such metering points do not yet have CPR numbers registered in DataHub. Therefore, it is not possible to use the flow described in section 1.6.3. CPR numbers are typically added in DataHub when a customer moves or changes electricity supplier.

Path: /api/MeteringPoints/MeteringPoint/Relation/Add/{meteringPointId}/{webAccessCode}
Parameters:

Name	Type	Value
meteringPointId	string	Id of the metering point
webAccessCode	string	Web access code for metering point

HTTP verb: PUT
Response: String with the value of the metering point id. See example in appendix B (ii. Get data access token / Add relation based on CPR/CVR / Add relation with web access code / Delete relation – String).

1.6.5 Delete relation

This request is used for deleting an existing relation to a metering point.

Path: /api/MeteringPoints/MeteringPoint/Relation/{meteringPointId}
Parameters:

Name	Type	Value
meteringPointId	string	Id of the metering point

HTTP verb: DELETE
Response: String with the value of the metering point id. See example in appendix B (ii. Get data access token / Add relation based on CPR/CVR / Add relation with web access code / Delete relation – String).

1.6.6 Get metering point details

This request is used for querying details (master data) for one or more (linked/related) metering point.

Path: /api/MeteringPoints/MeteringPoint/GetDetails
Parameters:

Name	Type	Value
meteringPointIds	String array	List of metering point ids See example in appendix B (i. List of metering points)

HTTP verb: POST
Response: See example in appendix B (iv. Get metering point details – MeteringPointDetailsPublicApi)

1.6.7 Get charges

This request is used for querying charge data (subscriptions, tariffs and fees) for one or more (linked/related) metering points. Charges linked to the metering point at the time of the request or on any future date will be returned.

Path: /api/MeteringPoints/MeteringPoint/GetCharges
Parameters:

Name	Type	Value
meteringPointIds	String array	List of metering point ids See example in appendix B (i. List of metering points)

HTTP verb: POST
Response: See example in appendix B (v. Get charges – MeteringPointPriceDataApi)

1.6.8 Get time series

This request is used for querying time series for one or more (linked/related) metering points for a specified period and with a specified aggregation level.

Path: /api/MeterData/GetTimeSeries/{dateFrom}/{dateTo}/{aggregation}
Parameters:

Name	Type	Value
meteringPointIds	String array	List of metering point ids See example in appendix B (i. List of metering points)
dateFrom	string	YYYY-MM-DD
dateTo	string	YYYY-MM-DD
aggregation	string	Actual | Quarter | Hour | Day | Month | Year

HTTP verb: POST
Response: See example in appendix B (vi. Get time series – MyEnergyDataMarketDocumentResponse)
Response details: The following table provides information about some of the main structures in the *Get time series* response.

Structure	Description
MyEnergyData_MarketDocument	One MyEnergyData_MarketDocument structure will be returned per metering point.
period.timeInterval	The structure specifies the total time interval for all time series in the specific MarketDocument structure.
TimeSeries	If the metering point is non-profiled settled or flex settled (settlementMethod = E02 or D01), a maximum of 1 TimeSeries structure may be returned, containing all returned non-profiled energy quantities. If the metering point is profiled settled (settlementMethod = E01), a maximum of 2 TimeSeries structures may be returned – one structure containing non-profiled energy quantities and another containing profiled energy quantities.
MarketEvaluationPoint	This structure specifies the location where the time series are measured.
Period	For non-profiled energy quantities, the following applies: • 1 period per day will be returned if 15 minutes, hourly or daily resolution is returned • 1 period per month will be returned if monthly resolution is returned) • 1 period per year will be returned if yearly resolution is returned) For profiled energy quantities one period will be returned per energy quantity as registered in DataHub. Various period resolutions may apply.
Point	This structure contains 1-96 positions depending on the nature of the periods as described above.

1.6.9 Get meter readings

This request is used for querying meter readings for one or more (linked/related) metering points for a specified period.

Path: /api/MeterData/GetMeterReadings/{dateFrom}/{dateTo}

Parameters:

Name	Type	Value
meteringPointIds	String array	List of metering point ids See example in appendix B (i. List of metering points)
dateFrom	string	YYYY-MM-DD
dateTo	string	YYYY-MM-DD

HTTP verb: POST

Response: See example in appendix B (vii. Get meter readings – MeterDataReadingsPublicAPI)

2. Third party API

2.1 Introduction

This API enables third parties to access customers' data from DataHub based on authorization (power of attorney) granted by the customer. To gain access to the API the third party must:

- be registered as a third party to DataHub (further information can be found on Energinet's website: https://energinet.dk/El/Elmarkedet/Saadan-bliver-du-tredjepart).
- be authorized by use of a token (see section 2.5).

Authentication and authorization is handled by using tokens. Further information can be found in section 2.5. When using the token all endpoints described in this chapter are accessible.

To retrieve actual data the third party must be authorized by customers to access data for specific metering points for specified periods of time. The third party must request access from the customer by a separate process which is not part of the API. [More information about this process will follow later. However, the process is expected to be similar to the existing process, for which documentation can be found on the Energinet website (see link above).]

The following data can be requested by use of the third party API:

1. List of authorizations (power of attorneys) (see section 2.6.2)
2. List of metering points, including selected metering point details (master data) (see section 2.6.3)
3. Extended list of details (master data) per metering point (see section 2.6.4)
4. Charge data per metering point (see section 2.6.5)
5. Time series per metering point (see section 2.6.6)
6. Meter readings per metering point (see section 2.6.7)

2.2 API location

Environment	URL
Pre-production environment	https://apipreprod.eloverblik.dk/ThirdPartyApi/
Production environment	https://api.eloverblik.dk/ThirdPartyApi/

2.3 Correlation id

It is possible to set a correlation id in the request header using the 'X-User-Correlation-ID' (with an UUID). When provided this id will follow the request and finally end in the response header. This id can be used for tracking the request. Parallel to this id there is another internally id (Also a UUID), which also is returned in the response header as 'X-Correlation-ID'.

Please note that the *Get time series* response is subject to a different market message standard. Therefore, only one id (mRID) is returned in this response. This id is an internal id similar to the X-Correlation-ID.

2.4 Error codes and HTTP responses

A list of relevant error codes and HTTP status codes can be found in appendix A. Please note that the codes are subject to change.

2.5 Tokens

Authentication and authorization are handled by using OAuth 2.0 tokens. To get started a refresh token is required. A refresh token for third party API access can be created in the

Eloverblik portal after logging in with NemID as a registered third party. The token is a long text string (JWT token), which must be copied and stored for use with the system that needs to access the API.

When a refresh token is obtained, the token endpoint can be accessed to create a short-lived data access token (valid for 60 minutes). See section 2.6.1 for further information. For all data access the data access token needs to be supplied in the HTTP header:

Authorization: Bearer <data-access-token>

2.6 Endpoints

2.6.1 Get data access token

Path: /api/Token
Parameters: None
HTTP verb: GET
Response: See example in appendix B (ix. Get data access token – String)

Note: Include the refresh token in the HTTP header as mentioned in section 2.5.

2.6.2 Get authorizations

This request is used for getting details about authorizations (power of attorneys) granted by customers. Only data regarding valid/active authorizations are returned. Data regarding deleted or expired authorizations are not returned.

Path: /api/Authorization/Authorizations
Parameters: None
HTTP verb: GET
Response: See example in appendix B (x. Get authorizations – Authorization)

2.6.3 Get metering points

This request is used for getting a list of metering points to which access has been granted. A filter value must be supplied to limit the result.
The following filter options are available:

- authorizationId: Returns a list of metering points included in a specific customer authorization
- customerCVR: Returns a list of metering points to which access has been granted by a specific customer (based on the CVR of the customer/granting party)
- customerKey: Returns a list of metering points which are linked to a specific customer key. The customer key can be supplied by the third party in the process when the customer is requested to grant access.

Path: /api/Authorization/Authorization/MeteringPoints/{scope}/{identifier}
Parameters:

Name	Type	Value
scope	string	authorizationId | customerCVR | customerKey
identifier	string	Authorization Id, customer CVR or customer key

HTTP verb: GET

Response: See example in appendix B (xi. Get metering points – MeteringPointPublicApi)

2.6.4 Get metering point details

This request is used for querying details (master data) for one or more metering points (to which access has been granted).

Path: /api/MeteringPoint/GetDetails
Parameters:

Name	Type	Value
meteringPointIds	String array	List of metering point ids See example in appendix B (viii. List of metering points)

HTTP verb: POST
Response: See example in appendix B (xii. Get metering point details – MeteringPointDetailsPublicApi)

2.6.5 Get charges

This request is used for querying charge data (subscriptions and tariffs) for one or more metering points (to which access has been granted). Charges linked to the metering point at the time of the request or on any future date will be returned.

Path: /api/MeteringPoint/GetCharges
Parameters:

Name	Type	Value
meteringPointIds	String array	List of metering point ids See example in appendix B (viii. List of metering points)

HTTP verb: POST
Response: See example in appendix B (xiii. Get charges – MeteringPointPriceDataApi)

2.6.6 Get time series

This request is used for querying time series for one or more metering points (to which access has been granted) for a specified period and with a specified aggregation level.

Path: /api/MeterData/GetTimeSeries/{dateFrom}/{dateTo}/{aggregation}
Parameters:

Name	Type	Value
dateFrom	string	YYYY-MM-DD
dateTo	string	YYYY-MM-DD
aggregation	string	Actual | Quarter | Hour | Day | Month | Year
meteringPointIds	String array	List of metering point ids See example in appendix B (viii. List of metering points)

HTTP verb: POST
Response: See example in appendix B (xiv. Get time series – MyEnergyDataMarketDocumentResponse)
Response details: The following table provides information about some of the main structures in the *Get time series* response.

Structure	Description
MyEnergyData_MarketDocument	One MyEnergyData_MarketDocument structure will be returned per metering point.
period.timeInterval	The structure specifies the total time interval for all time series in the specific MarketDocument structure.
TimeSeries	If the metering point is non-profiled settled or flex settled (settlementMethod = E02 or D01), a maximum of 1 TimeSeries structure may be returned, containing all returned non-profiled energy quantities. If the metering point is profiled settled (settlementMethod = E01), a maximum of 2 TimeSeries structures may be returned – one structure containing non-profiled energy quantities and another containing profiled energy quantities.
MarketEvaluationPoint	This structure specifies the location where the time series are measured.
Period	For non-profiled energy quantities, the following applies: • 1 period per day will be returned if 15 minutes, hourly or daily resolution is returned • 1 period per month will be returned if monthly resolution is returned) • 1 period per year will be returned if yearly resolution is returned) For profiled energy quantities one period will be returned per energy quantity as registered in DataHub. Various period resolutions may apply.
Point	This structure contains 1-96 positions depending on the nature of the periods as described above.

2.6.7 Get meter readings

This request is used for querying meter readings for one or more metering points (to which access has been granted) for a specified period.

Path: /api/MeterData/GetMeterReadings/{dateFrom}/{dateTo}
Parameters:

Name	Type	Value
dateFrom	string	YYYY-MM-DD
dateTo	string	YYYY-MM-DD
meteringPointIds	String array	List of metering point ids See example in appendix B (viii. List of metering points)

HTTP verb: POST
Response: See example in appendix B (xv. Get meter readings – MeterDataReadingsPublicAPI)

Appendix A – Error codes and HTTP responses

Error code	Error code message	HTTP status code	HTTP message
10000	NoError	200	No errors
10001	WrongNumberOfArguments	400	Wrong number of arguments.
10002	ToManyRequestItems	412	To many request items.
10003	InternalServerError	500	Internal server error.
10004	MaximumNumberOfMeteringPointsExceeded	429	Number of metering point exceeds maximum of {max} metering points per request.
20000	WrongMeteringPointIdOrWebAccessCode	404	Invalid meteringpoint ID or webaccess code.
20001	MeteringPointBlocked	403	Meteringpoint blocked.
20002	MeteringPointAlreadyAdded	208	Meteringpoint relation already added.
20003	MeteringPointIdNot18CharsLong	411	Meteringpoint ID must be 18 characters long.
20004	MeteringpointIdContainsNonDigits	406	Meteringpoint IDs do not contain digits.
20005	MeteringPointAliasTooLong	416	Meteringpoint alias too long.
20006	WebAccessCodeNot8CharsLong	411	Webaccess codes must be 8 characters long.
20007	WebAccessCodeContainsIllegalChars	406	Webaccess code contains illegal characters.
20008	MeteringPointNotFound	404	Meteringpoint not found.
20009	MeteringPointIsChild	422	Meteringpoint can't be child.
20010	RelationNotFound	404	Relation not found.
20011	UnknownError	500	Unknown error
20012	Unauthorized	401	Unauthorized access.
20013	NoValidMeteringPointsInList	400	No meteringpoints in request conforms to valid meteringpoint format.
30000	FromDateIsGreaterThanToday	400	Requested from date is after today.
30001	FromDateIsGreaterThanToDate	400	Period not allowed, ToDate is before FromDate.
30002	ToDateCanNotBeEqualToFromDate	400	Period not allowed, ToDate is equal to FromDate
30003	ToDateIsGreaterThanToday	400	Requested to date is after today.
30004	InvalidDateFormat	400	Invalid date format in request.
30005	InvalidRequestParameters	400	A request parameter is invalid.
30006	AccessToMeteringPointDenied	401	Access to meterpoint denied.
30007	NoMeteringPointDataAviliable	204	No meterpoint data aviliable.
30008	RequestedAggregationUnavaliable	406	Requested data aggregation is not supported.
30009	InvalidMeteringpointId	406	Requested meteringpoint ID is not valid.
30010	DateNotCoveredByAuthorization	401	Requested date not covered by Authorization.
30011	AggrationNotValid	406	Requested data aggregation is not supported.
30012	RequestToHuge	413	Request size too large.
40000	InvalidCVR	403	CVR is invalid.
40001	InvalidIncludeFutureMeteringPointsRelatedToCVR	404	Requested future meteringpoints related to CVR are invalid.
40002	InvalidMasterDataFields	417	Invalid master data fields.
40003	InvalidMeteringPointIds	406	Requested meteringpoint IDs are not valid.
40004	InvalidSignature	403	Invalid signature.
40005	InvalidSignedByNameId	403	Invalid signed by name ID
40006	InvalidSignedDate	403	Invalid signed date.
40007	InvalidSignedText	403	Invalid signed text.
40008	InvalidThirdPartyId	403	Invalid third party ID.

40009	InvalidValidFrom	400	Invalid from date.
40010	InvalidValidTo	400	Invalid to date.
40011	ValidToBeforeValidFrom	400	Requested from date cannot be after requested to date.
40012	ValidToOutOfRange	400	Requested to date is out of range.
40013	ValidFromOutOfRange	400	Requested from date is out of range.
40014	NoAuthorizationsFound	400	No power of attorneys found.
50000	WrongTokenType	406	Request used wrong token type.
50001	TokenNotValid	401	Token is invalid.
50002	ErrorCreatingToken	500	Error creating token.
50003	TokenRegistrationFailed	500	Token registration failed.
50004	TokenAlreadyActive	405	Token already active.
50005	TokenAlreadyDeactivated	405	Token already deactived.
50006	TokenMissingTokenId	401	Token do not contain a token id.
60000	ThirdPartyNotFound	404	Third party not found.
60001	ThirdPartyWasNotCreated	400	Third party not created.
60002	ThirdPartyAlreadyExist	208	Third party already exist.
60004	ThirdPartyApplictionInPrgress	400	Third party application is already in progress.
60005	ThirdPartyAlreadyExistButIsInactive	401	Third party already exist but is inactive.
60006	ThirdPartyAlreadyExistButIsRevoked	401	Third party already exist but access is revoked.

Appendix B – Examples

Customer API – Request examples

i. List of metering points

JSON:

```
{
  "meteringPoints": {
    "meteringPoint": [
      "123456789123456789",
      "001234567890123456"
    ]
  }
}
```

XML:

```
<?xml version="1.0" encoding="UTF-8"?>
<MeteringPointsRequest>
    <meteringPoints>
        <meteringPoint>123456789123456789</meteringPoint>
        <meteringPoint>001234567890123456</meteringPoint>
    </meteringPoints>
</MeteringPointsRequest>
```

Customer API – Response examples

ii. Get data access token / Add relation based on CPR/CVR / Add relation with web access code / Delete relation – String

JSON:

```
{
  "result": [
    {
      "result": "string",
      "success": true,
      "errorCode": 10000,
      "errorText": "string",
      "id": "string"
    }
  ]
}
```

XML:

```
<?xml version="1.0" encoding="UTF-8"?>
<ApiResponse[List[Response[String]]]>
    <result>
        <result>string</result>
        <success>true</success>
        <errorCode>10000</errorCode>
        <errorText>string</errorText>
        <id>string</id>
    </result>
</ApiResponse[List[Response[String]]]>
```

iii. Get metering points – MeteringPointPublicApi

JSON:


```
{
 "result": [
  {
   "meteringPointId": "string",
   "typeOfMP": "string",
   "balanceSupplierName": "string",
   "streetCode": "string",
   "streetName": "string",
   "buildingNumber": "string",
   "floorId": "string",
   "roomId": "string",
   "postcode": "string",
   "cityName": "string",
   "citySubDivisionName": "string",
   "municipalityCode": "string",
   "locationDescription": "string",
   "settlementMethod": "string",
   "meterReadingOccurrence": "string",
   "firstConsumerPartyName": "string",
   "secondConsumerPartyName": "string",
   "consumerCVR": "string",
   "dataAccessCVR": "string",
   "meterNumber": "string",
   "consumerStartDate": "string",
   "hasRelation": true,
   "childMeteringPoints": [
    {
     "meteringPointId": "string",
     "parentMeteringPointId": "string",
     "typeOfMP": "string",
     "meterReadingOccurrence": "string",
     "meterNumber": "string"
    }
   ]
  }
 ]
}
```


XML:


```
<?xml version="1.0" encoding="UTF-8"?>
<ApiResponse[List[MeteringPointPublicApi]]>
    <result>
        <meteringPointId>string</meteringPointId>
        <typeOfMP>string</typeOfMP>
        <balanceSupplierName>string</balanceSupplierName>
        <streetCode>string</streetCode>
        <streetName>string</streetName>
        <buildingNumber>string</buildingNumber>
        <floorId>string</floorId>
        <roomId>string</roomId>
        <postcode>string</postcode>
```

```
            <cityName>string</cityName>
            <citySubDivisionName>string</citySubDivisionName>
            <municipalityCode>string</municipalityCode>
            <locationDescription>string</locationDescription>
            <settlementMethod>string</settlementMethod>
            <meterReadingOccurrence>string</meterReadingOccurrence>
            <firstConsumerPartyName>string</firstConsumerPartyName>
            <secondConsumerPartyName>string</secondConsumerPartyName>
            <consumerCVR>string</consumerCVR>
            <dataAccessCVR>string</dataAccessCVR>
            <meterNumber>string</meterNumber>
            <consumerStartDate>string</consumerStartDate>
            <hasRelation>true</hasRelation>
            <childMeteringPoints>
                <meteringPointId>string</meteringPointId>
                <parentMeteringPointId>string</parentMeteringPointId>
                <typeOfMP>string</typeOfMP>
                <meterReadingOccurrence>string</meterReadingOccurrence>
                <meterNumber>string</meterNumber>
            </childMeteringPoints>
        </result>
    </ApiResponse[List[MeteringPointPublicApi]]>
```

iv. Get metering point details – MeteringPointDetailsPublicApi

JSON:

```
{
  "result": [
   {
    "result": {
     "meteringPointId": "string",
     "parentMeteringPointId": "string",
     "typeOfMP": "string",
     "energyTimeSeriesMeasureUnit": "string",
     "estimatedAnnualVolume": "string",
     "settlementMethod": "string",
     "meterNumber": "string",
     "gridOperatorName": "string",
     "meteringGridAreaIdentification": "string",
     "netSettlementGroup": "string",
     "physicalStatusOfMP": "string",
     "consumerCategory": "string",
     "powerLimitKW": "string",
     "powerLimitA": "string",
     "subTypeOfMP": "string",
     "productionObligation": "string",
     "mpCapacity": "string",
     "mpConnectionType": "string",
     "disconnectionType": "string",
     "product": "string",
     "consumerCVR": "string",
     "dataAccessCVR": "string",
     "consumerStartDate": "string",
```

```
      "meterReadingOccurrence": "string",
      "mpReadingCharacteristics": "string",
      "meterCounterDigits": "string",
      "meterCounterMultiplyFactor": "string",
      "meterCounterUnit": "string",
      "meterCounterType": "string",
      "balanceSupplierName": "string",
      "balanceSupplierStartDate": "string",
      "taxReduction": "string",
      "taxSettlementDate": "string",
      "mpRelationType": "string",
      "streetCode": "string",
      "streetName": "string",
      "buildingNumber": "string",
      "floorId": "string",
      "roomId": "string",
      "postcode": "string",
      "cityName": "string",
      "citySubDivisionName": "string",
      "municipalityCode": "string",
      "locationDescription": "string",
      "firstConsumerPartyName": "string",
      "secondConsumerPartyName": "string",
      "contactAddresses": [
       {
        "contactName1": "string",
        "contactName2": "string",
        "addressCode": "string",
        "streetName": "string",
        "buildingNumber": "string",
        "floorId": "string",
        "roomId": "string",
        "citySubDivisionName": "string",
        "postcode": "string",
        "cityName": "string",
        "countryName": "string",
        "contactPhoneNumber": "string",
        "contactMobileNumber": "string",
        "contactEmailAddress": "string",
        "contactType": "string"
       }
      ],
      "childMeteringPoints": [
       {
        "meteringPointId": "string",
        "parentMeteringPointId": "string",
        "typeOfMP": "string",
        "meterReadingOccurrence": "string",
        "meterNumber": "string"
       }
      ]
     },
```

```
      "success": true,
      "errorCode": 10000,
      "errorText": "string",
      "id": "string"
    }
  ]
}
```

XML:

```
<?xml version="1.0" encoding="UTF-8"?>
<ApiResponse[List[Response[MeteringPointDetailsPublicApi]]]>
    <result>
        <result>
            <meteringPointId>string</meteringPointId>
            <parentMeteringPointId>string</parentMeteringPointId>
            <typeOfMP>string</typeOfMP>
            <energyTimeSeriesMeasureUnit>string</energyTimeSeriesMeasureUnit>
            <estimatedAnnualVolume>string</estimatedAnnualVolume>
            <settlementMethod>string</settlementMethod>
            <meterNumber>string</meterNumber>
            <gridOperatorName>string</gridOperatorName>
            <meteringGridAreaIdentification>string</meteringGridAreaIdentification>
            <netSettlementGroup>string</netSettlementGroup>
            <physicalStatusOfMP>string</physicalStatusOfMP>
            <consumerCategory>string</consumerCategory>
            <powerLimitKW>string</powerLimitKW>
            <powerLimitA>string</powerLimitA>
            <subTypeOfMP>string</subTypeOfMP>
            <productionObligation>string</productionObligation>
            <mpCapacity>string</mpCapacity>
            <mpConnectionType>string</mpConnectionType>
            <disconnectionType>string</disconnectionType>
            <product>string</product>
            <consumerCVR>string</consumerCVR>
            <dataAccessCVR>string</dataAccessCVR>
            <consumerStartDate>string</consumerStartDate>
            <meterReadingOccurrence>string</meterReadingOccurrence>
            <mpReadingCharacteristics>string</mpReadingCharacteristics>
            <meterCounterDigits>string</meterCounterDigits>
            <meterCounterMultiplyFactor>string</meterCounterMultiplyFactor>
            <meterCounterUnit>string</meterCounterUnit>
            <meterCounterType>string</meterCounterType>
            <balanceSupplierName>string</balanceSupplierName>
            <balanceSupplierStartDate>string</balanceSupplierStartDate>
            <taxReduction>string</taxReduction>
            <taxSettlementDate>string</taxSettlementDate>
            <mpRelationType>string</mpRelationType>
            <streetCode>string</streetCode>
            <streetName>string</streetName>
            <buildingNumber>string</buildingNumber>
            <floorId>string</floorId>
            <roomId>string</roomId>
            <postcode>string</postcode>
```

```
            <cityName>string</cityName>
            <citySubDivisionName>string</citySubDivisionName>
            <municipalityCode>string</municipalityCode>
            <locationDescription>string</locationDescription>
            <firstConsumerPartyName>string</firstConsumerPartyName>
            <secondConsumerPartyName>string</secondConsumerPartyName>
            <contactAddresses>
                <contactName1>string</contactName1>
                <contactName2>string</contactName2>
                <addressCode>string</addressCode>
                <streetName>string</streetName>
                <buildingNumber>string</buildingNumber>
                <floorId>string</floorId>
                <roomId>string</roomId>
                <citySubDivisionName>string</citySubDivisionName>
                <postcode>string</postcode>
                <cityName>string</cityName>
                <countryName>string</countryName>
                <contactPhoneNumber>string</contactPhoneNumber>
                <contactMobileNumber>string</contactMobileNumber>
                <contactEmailAddress>string</contactEmailAddress>
                <contactType>string</contactType>
            </contactAddresses>
            <childMeteringPoints>
                <meteringPointId>string</meteringPointId>
                <parentMeteringPointId>string</parentMeteringPointId>
                <typeOfMP>string</typeOfMP>
                <meterReadingOccurrence>string</meterReadingOccurrence>
                <meterNumber>string</meterNumber>
            </childMeteringPoints>
        </result>
        <success>true</success>
        <errorCode>10000</errorCode>
        <errorText>string</errorText>
        <id>string</id>
    </result>
</ApiResponse[List[Response[MeteringPointDetailsPublicApi]]]>
```

v. Get charges – MeteringPointPriceDataApi

JSON:

```
{
  "result": [
   {
    "result": {
     "meteringPointId": "string",
     "subscriptions": [
      {
       "subscriptionId": "string",
       "name": "string",
       "description": "string",
       "owner": "string",
       "validFromDate": "string",
```

```
            "validToDate": "string",
            "price": 0,
            "quantity": 0
           }
          ],
          "fees": [
           {
            "feeId": "string",
            "name": "string",
            "description": "string",
            "owner": "string",
            "validFromDate": "string",
            "validToDate": "string",
            "price": 0,
            "quantity": 0
           }
          ],
          "tariffs": [
           {
            "tariffId": "string",
            "name": "string",
            "description": "string",
            "owner": "string",
            "periodType": "string",
            "validFromDate": "string",
            "validToDate": "string",
            "prices": [
             {
              "position": "string",
              "price": 0
             }
            ]
           }
          ]
         },
         "success": true,
         "errorCode": 10000,
         "errorText": "string",
         "id": "string"
        }
       ]
      }
```


XML:


```
<?xml version="1.0" encoding="UTF-8"?>
<ApiResponse[List[Response[MeteringPointPriceDataApi]]]>
    <result>
        <result>
            <meteringPointId>string</meteringPointId>
            <subscriptions>
                <subscriptionId>string</subscriptionId>
                <name>string</name>
                <description>string</description>
```

```
                    <owner>string</owner>
                    <validFromDate>string</validFromDate>
                    <validToDate>string</validToDate>
                    <price>0</price>
                    <quantity>0</quantity>
                </subscriptions>
                <fees>
                    <feeId>string</feeId>
                    <name>string</name>
                    <description>string</description>
                    <owner>string</owner>
                    <validFromDate>string</validFromDate>
                    <validToDate>string</validToDate>
                    <price>0</price>
                    <quantity>0</quantity>
                </fees>
                <tariffs>
                    <tariffId>string</tariffId>
                    <name>string</name>
                    <description>string</description>
                    <owner>string</owner>
                    <periodType>string</periodType>
                    <validFromDate>string</validFromDate>
                    <validToDate>string</validToDate>
                    <prices>
                        <position>string</position>
                        <price>0</price>
                    </prices>
                </tariffs>
            </result>
            <success>true</success>
            <errorCode>10000</errorCode>
            <errorText>string</errorText>
            <id>string</id>
        </result>
    </ApiResponse[List[Response[MeteringPointPriceDataApi]]]>
```

vi. Get time series – MyEnergyDataMarketDocumentResponse

JSON:

```
{
  "result": [
   {
    "MyEnergyData_MarketDocument": {
     "mRID": "string",
     "createdDateTime": "string",
     "sender_MarketParticipant.name": "string",
     "sender_MarketParticipant.mRID": {
      "codingScheme": "string",
      "name": "string"
     },
     "period.timeInterval": {
      "start": "string",
```

```
          "end": "string"
        },
        "TimeSeries": [
         {
          "mRID": "string",
          "businessType": "string",
          "curveType": "string",
          "measurement_Unit.name": "string",
          "MarketEvaluationPoint": {
           "mRID": {
            "codingScheme": "string",
            "name": "string"
           }
          },
          "Period": [
           {
            "resolution": "string",
            "timeInterval": {
             "start": "string",
             "end": "string"
            },
            "Point": [
             {
              "position": "string",
              "out_Quantity.quantity": "string",
              "out_Quantity.quality": "string"
             }
            ]
           }
          ]
         }
        ],
        "success": true,
        "errorCode": "string",
        "errorText": "string",
        "id": "string",
        "stackTrace": "string"
       }
      ]
     }
```


XML:


```
<?xml version="1.0" encoding="UTF-8"?>
<ApiResponse[List[MyEnergyDataMarketDocumentResponse]]>
    <result>
        <MyEnergyData_MarketDocument>
            <mRID>string</mRID>
            <createdDateTime>string</createdDateTime>
            <sender_MarketParticipant.name>string</sender_MarketParticipant.name>
            <sender_MarketParticipant.mRID>
                <codingScheme>string</codingScheme>
                <name>string</name>
```

```
                </sender_MarketParticipant.mRID>
                <period.timeInterval>
                    <start>string</start>
                    <end>string</end>
                </period.timeInterval>
                <TimeSeries>
                    <mRID>string</mRID>
                    <businessType>string</businessType>
                    <curveType>string</curveType>
                    <measurement_Unit.name>string</measurement_Unit.name>
                    <MarketEvaluationPoint>
                        <mRID>
                            <codingScheme>string</codingScheme>
                            <name>string</name>
                        </mRID>
                    </MarketEvaluationPoint>
                    <Period>
                        <resolution>string</resolution>
                        <timeInterval>
                            <start>string</start>
                            <end>string</end>
                        </timeInterval>
                        <Point>
                            <position>string</position>
                            <out_Quantity.quantity>string</out_Quantity.quantity>
                            <out_Quantity.quality>string</out_Quantity.quality>
                        </Point>
                    </Period>
                </TimeSeries>
            </MyEnergyData_MarketDocument>
            <success>true</success>
            <errorCode>string</errorCode>
            <errorText>string</errorText>
            <id>string</id>
            <stackTrace>string</stackTrace>
        </result>
    </ApiResponse[List[MyEnergyDataMarketDocumentResponse]]>
```

vii. Get meter readings – MeterDataReadingsPublicAPI

JSON:

```
{
  "result": [
   {
    "result": {
     "meteringPointId": "string",
     "readings": [
      {
       "readingDate": "string",
       "registrationDate": "string",
       "meterNumber": "string",
       "meterReading": "string",
       "measurementUnit": "string"
```

```
        }
       ]
      },
      "success": true,
      "errorCode": "string",
      "errorText": "string",
      "id": "string",
      "stackTrace": "string"
     }
    ]
   }
```


XML:


```
<?xml version="1.0" encoding="UTF-8"?>
<ApiResponse[List[Response[MeterDataReadingsPublicAPI]]]>
    <result>
        <result>
            <meteringPointId>string</meteringPointId>
            <readings>
                <readingDate>string</readingDate>
                <registrationDate>string</registrationDate>
                <meterNumber>string</meterNumber>
                <meterReading>string</meterReading>
                <measurementUnit>string</measurementUnit>
            </readings>
        </result>
        <success>true</success>
        <errorCode>string</errorCode>
        <errorText>string</errorText>
        <id>string</id>
        <stackTrace>string</stackTrace>
    </result>
</ApiResponse[List[Response[MeterDataReadingsPublicAPI]]]>
```


Third party API – Request examples

viii. List of metering points

JSON:


```
{
 "meteringPoints": {
  "meteringPoint": [
   "123456789123456789",
   "001234567890123456"
  ]
 }
}
```


XML:


```
<?xml version="1.0" encoding="UTF-8"?>
<MeteringPointsRequest>
    <meteringPoints>
        <meteringPoint>123456789123456789</meteringPoint>
        <meteringPoint>001234567890123456</meteringPoint>
```

```
        </meteringPoints>
    </MeteringPointsRequest>
```

Third party API – Successful response examples

ix. Get data access token – String

JSON:


```
{
  "result": [
    {
      "result": "string",
      "success": true,
      "errorCode": 10000,
      "errorText": "string",
      "id": "string"
    }
  ]
}
```


XML:


```
<?xml version="1.0" encoding="UTF-8"?>
<ApiResponse[List[Response[String]]]>
    <result>
        <result>string</result>
        <success>true</success>
        <errorCode>10000</errorCode>
        <errorText>string</errorText>
        <id>string</id>
    </result>
</ApiResponse[List[Response[String]]]>
```


x. Get authorizations – Authorization

JSON:


```
{
  "result": [
    {
      "id": "string",
      "thirdPartyName": "string",
      "validFrom": "string",
      "validTo": "string",
      "customerName": "string",
      "customerCVR": "string",
      "customerKey": "string",
      "includeFutureMeteringPoints": true,
      "timeStamp": "2019-04-15T09:49:06.642Z"
    }
  ]
}
```


XML:

```
<?xml version="1.0" encoding="UTF-8"?>
<ApiResponse[List[Authorization]]>
    <result>
        <id>string</id>
        <thirdPartyName>string</thirdPartyName>
        <validFrom>string</validFrom>
        <validTo>string</validTo>
        <customerName>string</customerName>
        <customerCVR>string</customerCVR>
        <customerKey>string</customerKey>
        <includeFutureMeteringPoints>true</includeFutureMeteringPoints>
        <timeStamp>2019-04-15T09:55:15.175Z</timeStamp>
    </result>
</ApiResponse[List[Authorization]]>
```

xi. Get metering points – MeteringPointPublicApi

JSON:

```
{
 "result": [
  {
   "meteringPointId": "string",
   "typeOfMP": "string",
   "accessFrom": "string",
   "accessTo": "string",
   "streetCode": "string",
   "streetName": "string",
   "buildingNumber": "string",
   "floorId": "string",
   "roomId": "string",
   "postcode": "string",
   "cityName": "string",
   "citySubDivisionName": "string",
   "municipalityCode": "string",
   "locationDescription": "string",
   "settlementMethod": "string",
   "meterReadingOccurrence": "string",
   "firstConsumerPartyName": "string",
   "secondConsumerPartyName": "string",
   "consumerCVR": "string",
   "dataAccessCVR": "string",
   "meterNumber": "string",
   "consumerStartDate": "string",
   "childMeteringPoints": [
    {
     "meteringPointId": "string",
     "parentMeteringPointId": "string",
     "typeOfMP": "string",
     "meterReadingOccurrence": "string",
     "meterNumber": "string"
    }
   ]
  }
```

```
    ]
  }
```

XML:

```
  <?xml version="1.0" encoding="UTF-8"?>
  <ApiResponse[List[MeteringPointPublicApi]]>
    <result>
      <meteringPointId>string</meteringPointId>
      <typeOfMP>string</typeOfMP>
      <accessFrom>string</accessFrom>
      <accessTo>string</accessTo>
      <streetCode>string</streetCode>
      <streetName>string</streetName>
      <buildingNumber>string</buildingNumber>
      <floorId>string</floorId>
      <roomId>string</roomId>
      <postcode>string</postcode>
      <cityName>string</cityName>
      <citySubDivisionName>string</citySubDivisionName>
      <municipalityCode>string</municipalityCode>
      <locationDescription>string</locationDescription>
      <settlementMethod>string</settlementMethod>
      <meterReadingOccurrence>string</meterReadingOccurrence>
      <firstConsumerPartyName>string</firstConsumerPartyName>
      <secondConsumerPartyName>string</secondConsumerPartyName>
      <consumerCVR>string</consumerCVR>
      <dataAccessCVR>string</dataAccessCVR>
      <meterNumber>string</meterNumber>
      <consumerStartDate>string</consumerStartDate>
      <childMeteringPoints>
        <meteringPointId>string</meteringPointId>
        <parentMeteringPointId>string</parentMeteringPointId>
        <typeOfMP>string</typeOfMP>
        <meterReadingOccurrence>string</meterReadingOccurrence>
        <meterNumber>string</meterNumber>
      </childMeteringPoints>
    </result>
  </ApiResponse[List[MeteringPointPublicApi]]>
```

xii. Get metering point details – MeteringPointDetailsPublicApi

JSON:

```
  {
   "result": [
    {
     "result": {
      "meteringPointId": "string",
      "parentMeteringPointId": "string",
      "typeOfMP": "string",
      "energyTimeSeriesMeasureUnit": "string",
      "estimatedAnnualVolume": "string",
      "settlementMethod": "string",
      "meterNumber": "string",
      "gridOperatorName": "string",
```

```
"meteringGridAreaIdentification": "string",
"netSettlementGroup": "string",
"physicalStatusOfMP": "string",
"consumerCategory": "string",
"powerLimitKW": "string",
"powerLimitA": "string",
"subTypeOfMP": "string",
"productionObligation": "string",
"mpCapacity": "string",
"mpConnectionType": "string",
"disconnectionType": "string",
"product": "string",
"consumerCVR": "string",
"dataAccessCVR": "string",
"consumerStartDate": "string",
"meterReadingOccurrence": "string",
"mpReadingCharacteristics": "string",
"meterCounterDigits": "string",
"meterCounterMultiplyFactor": "string",
"meterCounterUnit": "string",
"meterCounterType": "string",
"taxReduction": "string",
"taxSettlementDate": "string",
"mpRelationType": "string",
"streetCode": "string",
"streetName": "string",
"buildingNumber": "string",
"floorId": "string",
"roomId": "string",
"postcode": "string",
"cityName": "string",
"citySubDivisionName": "string",
"municipalityCode": "string",
"locationDescription": "string",
"firstConsumerPartyName": "string",
"secondConsumerPartyName": "string",
"contactAddresses": [
 {
   "contactName1": "string",
   "contactName2": "string",
   "addressCode": "string",
   "streetName": "string",
   "buildingNumber": "string",
   "floorId": "string",
   "roomId": "string",
   "citySubDivisionName": "string",
   "postcode": "string",
   "cityName": "string",
   "countryName": "string",
   "contactPhoneNumber": "string",
   "contactMobileNumber": "string",
   "contactEmailAddress": "string",
```

```
            "contactType": "string"
           }
          ],
          "childMeteringPoints": [
           {
            "meteringPointId": "string",
            "parentMeteringPointId": "string",
            "typeOfMP": "string",
            "meterReadingOccurrence": "string",
            "meterNumber": "string"
           }
          ]
         },
         "success": true,
         "errorCode": 10000,
         "errorText": "string",
         "id": "string"
        }
       ]
      }
```

XML:

```
<?xml version="1.0" encoding="UTF-8"?>
<ApiResponse[List[Response[MeteringPointDetailsPublicApi]]]>
    <result>
        <result>
            <meteringPointId>string</meteringPointId>
            <parentMeteringPointId>string</parentMeteringPointId>
            <typeOfMP>string</typeOfMP>
            <energyTimeSeriesMeasureUnit>string</energyTimeSeriesMeasureUnit>
            <estimatedAnnualVolume>string</estimatedAnnualVolume>
            <settlementMethod>string</settlementMethod>
            <meterNumber>string</meterNumber>
            <gridOperatorName>string</gridOperatorName>
            <meteringGridAreaIdentification>string</meteringGridAreaIdentification>
            <netSettlementGroup>string</netSettlementGroup>
            <physicalStatusOfMP>string</physicalStatusOfMP>
            <consumerCategory>string</consumerCategory>
            <powerLimitKW>string</powerLimitKW>
            <powerLimitA>string</powerLimitA>
            <subTypeOfMP>string</subTypeOfMP>
            <productionObligation>string</productionObligation>
            <mpCapacity>string</mpCapacity>
            <mpConnectionType>string</mpConnectionType>
            <disconnectionType>string</disconnectionType>
            <product>string</product>
            <consumerCVR>string</consumerCVR>
            <dataAccessCVR>string</dataAccessCVR>
            <consumerStartDate>string</consumerStartDate>
            <meterReadingOccurrence>string</meterReadingOccurrence>
            <mpReadingCharacteristics>string</mpReadingCharacteristics>
            <meterCounterDigits>string</meterCounterDigits>
            <meterCounterMultiplyFactor>string</meterCounterMultiplyFactor>
```

```
            <meterCounterUnit>string</meterCounterUnit>
            <meterCounterType>string</meterCounterType>
            <taxReduction>string</taxReduction>
            <taxSettlementDate>string</taxSettlementDate>
            <mpRelationType>string</mpRelationType>
            <streetCode>string</streetCode>
            <streetName>string</streetName>
            <buildingNumber>string</buildingNumber>
            <floorId>string</floorId>
            <roomId>string</roomId>
            <postcode>string</postcode>
            <cityName>string</cityName>
            <citySubDivisionName>string</citySubDivisionName>
            <municipalityCode>string</municipalityCode>
            <locationDescription>string</locationDescription>
            <firstConsumerPartyName>string</firstConsumerPartyName>
            <secondConsumerPartyName>string</secondConsumerPartyName>
            <contactAddresses>
                <contactName1>string</contactName1>
                <contactName2>string</contactName2>
                <addressCode>string</addressCode>
                <streetName>string</streetName>
                <buildingNumber>string</buildingNumber>
                <floorId>string</floorId>
                <roomId>string</roomId>
                <citySubDivisionName>string</citySubDivisionName>
                <postcode>string</postcode>
                <cityName>string</cityName>
                <countryName>string</countryName>
                <contactPhoneNumber>string</contactPhoneNumber>
                <contactMobileNumber>string</contactMobileNumber>
                <contactEmailAddress>string</contactEmailAddress>
                <contactType>string</contactType>
            </contactAddresses>
            <childMeteringPoints>
                <meteringPointId>string</meteringPointId>
                <parentMeteringPointId>string</parentMeteringPointId>
                <typeOfMP>string</typeOfMP>
                <meterReadingOccurrence>string</meterReadingOccurrence>
                <meterNumber>string</meterNumber>
            </childMeteringPoints>
        </result>
        <success>true</success>
        <errorCode>10000</errorCode>
        <errorText>string</errorText>
        <id>string</id>
    </result>
</ApiResponse[List[Response[MeteringPointDetailsPublicApi]]]>
```

xiii. Get charges – MeteringPointPriceDataApi

JSON:

```
{
```

```
  "result": [
   {
    "result": {
     "meteringPointId": "string",
     "subscriptions": [
      {
       "subscriptionId": "string",
       "name": "string",
       "description": "string",
       "owner": "string",
       "validFromDate": "string",
       "validToDate": "string",
       "price": 0,
       "quantity": 0
      }
     ],
     "tariffs": [
      {
       "tariffId": "string",
       "name": "string",
       "description": "string",
       "owner": "string",
       "periodType": "string",
       "validFromDate": "string",
       "validToDate": "string",
       "prices": [
        {
         "position": "string",
         "price": 0
        }
       ]
      }
     ]
    },
    "success": true,
    "errorCode": 10000,
    "errorText": "string",
    "id": "string"
   }
  ]
 }
```


XML:


```
<?xml version="1.0" encoding="UTF-8"?>
<ApiResponse[List[Response[MeteringPointPriceDataApi]]]>
    <result>
        <result>
            <meteringPointId>string</meteringPointId>
            <subscriptions>
                <subscriptionId>string</subscriptionId>
                <name>string</name>
                <description>string</description>
                <owner>string</owner>
```

```
                    <validFromDate>string</validFromDate>
                    <validToDate>string</validToDate>
                    <price>0</price>
                    <quantity>0</quantity>
                </subscriptions>
                <tariffs>
                    <tariffId>string</tariffId>
                    <name>string</name>
                    <description>string</description>
                    <owner>string</owner>
                    <periodType>string</periodType>
                    <validFromDate>string</validFromDate>
                    <validToDate>string</validToDate>
                    <prices>
                        <position>string</position>
                        <price>0</price>
                    </prices>
                </tariffs>
            </result>
            <success>true</success>
            <errorCode>10000</errorCode>
            <errorText>string</errorText>
            <id>string</id>
        </result>
    </ApiResponse[List[Response[MeteringPointPriceDataApi]]]>
```

xiv. Get time series – MyEnergyDataMarketDocumentResponse

JSON:

```
{
  "result": [
   {
    "MyEnergyData_MarketDocument": {
     "mRID": "string",
     "createdDateTime": "string",
     "sender_MarketParticipant.name": "string",
     "sender_MarketParticipant.mRID": {
      "codingScheme": "string",
      "name": "string"
     },
     "period.timeInterval": {
      "start": "string",
      "end": "string"
     },
     "TimeSeries": [
      {
       "mRID": "string",
       "businessType": "string",
       "curveType": "string",
       "measurement_Unit.name": "string",
       "MarketEvaluationPoint": {
        "mRID": {
         "codingScheme": "string",
```

```
            "name": "string"
           }
          },
          "Period": [
           {
            "resolution": "string",
            "timeInterval": {
             "start": "string",
             "end": "string"
            },
            "Point": [
             {
              "position": "string",
              "out_Quantity.quantity": "string",
              "out_Quantity.quality": "string"
             }
            ]
           }
          ]
         }
        ],
        "success": true,
        "errorCode": "string",
        "errorText": "string",
        "id": "string",
        "stackTrace": "string"
       }
      ]
     }
```


XML:


```
<?xml version="1.0" encoding="UTF-8"?>
<ApiResponse[List[MyEnergyDataMarketDocumentResponse]]>
    <result>
        <MyEnergyData_MarketDocument>
            <mRID>string</mRID>
            <createdDateTime>string</createdDateTime>
            <sender_MarketParticipant.name>string</sender_MarketParticipant.name>
            <sender_MarketParticipant.mRID>
                <codingScheme>string</codingScheme>
                <name>string</name>
            </sender_MarketParticipant.mRID>
            <period.timeInterval>
                <start>string</start>
                <end>string</end>
            </period.timeInterval>
            <TimeSeries>
                <mRID>string</mRID>
                <businessType>string</businessType>
                <curveType>string</curveType>
                <measurement_Unit.name>string</measurement_Unit.name>
                <MarketEvaluationPoint>
```

```
                        <mRID>
                            <codingScheme>string</codingScheme>
                            <name>string</name>
                        </mRID>
                    </MarketEvaluationPoint>
                    <Period>
                        <resolution>string</resolution>
                        <timeInterval>
                            <start>string</start>
                            <end>string</end>
                        </timeInterval>
                        <Point>
                            <position>string</position>
                            <out_Quantity.quantity>string</out_Quantity.quantity>
                            <out_Quantity.quality>string</out_Quantity.quality>
                        </Point>
                    </Period>
                </TimeSeries>
            </MyEnergyData_MarketDocument>
            <success>true</success>
            <errorCode>string</errorCode>
            <errorText>string</errorText>
            <id>string</id>
            <stackTrace>string</stackTrace>
        </result>
    </ApiResponse[List[MyEnergyDataMarketDocumentResponse]]>
```

xv. Get meter readings – MeterDataReadingsPublicAPI

JSON:

```
{
  "result": [
    {
      "result": {
        "meteringPointId": "string",
        "readings": [
          {
            "readingDate": "string",
            "registrationDate": "string",
            "meterNumber": "string",
            "meterReading": "string",
            "measurementUnit": "string"
          }
        ]
      },
      "success": true,
      "errorCode": "string",
      "errorText": "string",
      "id": "string",
      "stackTrace": "string"
    }
  ]
}
```

XML:

```
<?xml version="1.0" encoding="UTF-8"?>
<ApiResponse[List[Response[MeterDataReadingsPublicAPI]]]>
    <result>
        <result>
            <meteringPointId>string</meteringPointId>
            <readings>
                <readingDate>string</readingDate>
                <registrationDate>string</registrationDate>
                <meterNumber>string</meterNumber>
                <meterReading>string</meterReading>
                <measurementUnit>string</measurementUnit>
            </readings>
        </result>
        <success>true</success>
        <errorCode>string</errorCode>
        <errorText>string</errorText>
        <id>string</id>
        <stackTrace>string</stackTrace>
    </result>
</ApiResponse[List[Response[MeterDataReadingsPublicAPI]]]>
```