Addendum No. 1 to RFP 24-49

CITY OF SOMERVILLE, MASSACHUSETTS
Department of Procurement and Contracting Services
KATJANA BALLANTYNE
MAYOR

To: All Parties on Record with the City of Somerville as Holding
RFP 24-49 Integration of 311's QAlert Platform with the TreeKeeper Platform

From: Thupten Chukhatsang, Senior Procurement Manager

Date: 3/5/2024

Re: Questions and Answers, Clarifications to Scope
Attachment of DRG's TreeKeeper API Guide Template

Addendum No. 1 to RFP 24-49

Please acknowledge receipt of this Addendum by signing below and including this form in your proposal package. Failure to do so may subject the proposer to disqualification.

NAME OF COMPANY / INDIVIDUAL: _____

ADDRESS:_____

CITY/STATE/ZIP:_____

TELEPHONE/FAX/EMAIL:_____

SIGNATURE OF AUTHORIZED INDIVIDUAL:_____

ACKNOWLEDGEMENT OF ADDENDA:

Addendum #1 _____ #2 _____ #3_____ #4 _____

Addendum No. 1 to RFP 24-49

#	Question	Answer/Clarifications
1.	Does the TreeKeeper Platform have an API available? If so, is any API documentation available for my review at this time?	The specific API for the Somerville TreeKeeper account is not yet created, but a generic version "DRG_TreeKeeper API Guide Template" has been attached. This generic version contains very general/ generic API instructions. The selected vendor will need to work with Davey Resource Group (DRG) to create the Somerville-specific documentation. DRG will create the specific documentation to assist the 3rd party integration after there is a an agreed upon scope of work with a defined budget of hours. It is expected that in order for DRG's dev team to customize the API guide for a specific client, an agreement of 5 hours of time @$150/hr is required. Vendor would be responsible for obtaining the API information.
2.	To meet Scope of Work 2-2.4, would any new UI elements need to be added to the TreeKeeper platform? (If so, this could only be done by Davey Resource Group)	The "Calls" feature already exists in the TreeKeeper software. Most of the necessary features already exist in the "Calls" environment. A few minor modifications may be necessary. At the request of the selected vendor, City staff will work with Davey Resource Group to complete any necessary modifications that only Davey Resource Group can do.
3.	Add the following information to the scope:	Davey Resource Group has stated that their GIS/IT Development team will charge $150 per hour for API support.
4.	Can Somerville provide API documentation for the QAlert 311 System for our review?	The API for QAlert can be downloaded by filling out the form at this website: https://developers.qscend.com/. While filling out the form, be sure to indicate that the request is in relation to Somerville RFP 24-29. After submitting the form you will get an email with login credentials, and will be able to access the API.

5.	What kind of support will QAlert provide during the development process? Are there any charges related to QAlert support?	The company that runs QAlert (now Catalis) will provide the robust API for QAlert, free of charge, and can provide ordinary support during the development process, such as answering questions. Anything requiring a developer's time or more than 30 minutes of a technicians time would be billed at an hourly rate, but the rate was not provided. Catalis cannot guarantee accepting any developer work, including any work to update the User Interface. However we do not intend to change the functionality of QAlert, or the User Interface, so we do not believe the work described in this RFP would require any adjustments to the QAlert User Interface.
6.	Regarding "3.2 Develop mechanisms for automatic notifications to constituents when there are updates or closures related to their service requests." Does QAlert include functionality that notifies a constituent when a change has been made to the service request? We could certainly set up a notification service within TK if QAlert is not able to do this. However, this doesn't feel intuitive to the constituent who made the original request in the QAlert platform... I guess I'm just wondering if this is already a feature of QAlert and if so, why is it in this RFP? Side note: in the "Deliverables" section on page 13 they specify that notifications should happen within the QAlert system. It might be good to establish if that is already active or if they would need to work with QAlert to set up notifications and if there are costs associated with that.	Yes, QAlert does provide notifications to the constituents. Although it is already a feature of QAlert, it is included in this RFP to ensure that the set up of the 2-way communication maintains the functionality of that feature.

Attachment (following pages):
DRG's TreeKeeper API Guide Template (27 pages)

Introduction

Davey Resource Group's TreeKeeper 8 / myROWKeeper system has various REST APIs that a customer can use to pull and push data into TreeKeeper 8 / myROWKeeper. Each of these APIs have different authentication methods as we move forward to unifying them under a common method. This document describes how to authenticate and utilize our APIs for the following reasons

1) Querying / Pulling Data from TreeKeeper 8 / myROWKeeper
2) Send Site Data to TreeKeeper 8 / myROWKeeper
3) Send Work Record Data TreeKeeper 8

Version: 2023.11.20

Querying / Pulling Data from TreeKeeper 8 / myROWKeeper

data.treekeepersoftware.com API Endpoints:

A customer will be provided with a Token, GroupKey, and reportID when making calls to this endpoint. A where clause can be provided to control the data that the system is serving on these calls. The parameters of this clause are dependent on the structure of the data contained in the report that is being served. In short, any attribute and be queried against. Below is an example of pulling data that has been edited on and since October 30th, 2023.

Pull Data Example

```
Unset
curl --request POST \
 --url https://data.treekeepersoftware.com/view \
 --header 'Authorization: Bearer XXX' \
 --header 'Content-Type: application/json' \
 --header 'Cookie:
dtCookie=v_4_srv_6_sn_27633F39035E14C1ACFAEDC739F4325F_perc_10000
0_ol_0_mul_1_app-3Aea7c4b59f27d43eb_1' \
 --cookie
dtCookie=v_4_srv_6_sn_27633F39035E14C1ACFAEDC739F4325F_perc_10000
0_ol_0_mul_1_app-3Aea7c4b59f27d43eb_1 \
 --data '{
  "groupkey": "XXX",
  "whereclause": [
    {
      "columnName": "inspect_dt",
      "value": "2023-10-30",
      "joinType": "and",
      "operatorType":">"
    }
  ],
  "reportid": XXX,
  "datatype": "json"
}''
```

Version: 2023.11.20

Switching to “Production” Calls

Swap the Token and GroupKey to the credentials documented below

Other report ids to reference:

Testing Report ID	Description
XXX	Example reports will be listed here with descriptive details

Supported data types are

- xlsx
- pdf
- html
- json
- csv

Supported whereclause joinTypes

- and

Supported whereclause operatorTypes

- `'='`
- `'!='`
- `'<>'`
- `'<'`
- `'>'`
- `'>='`
- `'<='`
- `'like'`
- `'not like'`
- `'similar to'`
- `'not similar to'`
- `'~'`
- `'~*'`
- `'!~'`
- `'!~*'`

Version: 2023.11.20

data.treekeepersoftware.com API Endpoints Authentication Information

Customer Name - Production
Client Key: XXXX
Groupkey: XXXX

Sending Data to TreeKeeper 8 / myROWKeeper

https://api.treekeepersoftware.com

Authentication/Authorization

Before using the API, you will need access to an Auth JWT token that denotes your user and project. Once you have that, you can include it in the request inside the following HTTP header.

Here is an example curl to get the token:

```
curl --request GET \
  --url
'https://api.treekeepersoftware.com/authentication_service/token?projectid=
XXX&app=XXX' \
  --header 'Authorization: Basic XXX' \
  --cookie
dtCookie=v_4_srv_1_sn_D28B3CDCC92313F57EE1BFF441B84D37_perc_100000_ol_0_mul
_1_app-3Aea7c4b59f27d43eb_1_rcs-3Acss_0
```

Details needed for this call are…

Projectid = XXX
App = XXX
Username = XXX
Password = XXX

Once you receive this token, it will need to be included in all subsequent calls

Version: 2023.11.20

```
Unset
Authorization: Bearer -jwt token here-
```

Upserting Data TreeKeeperSoftware (https://api.treekeepersoftware.com/sites/features)

In order to upsert data to the system, a specific json body must be constructed and sent. Here is an example curl call:

```
curl --request POST \
  --url https://api.treekeepersoftware.com/sites/features \
  --header 'Authorization: Bearer -jwt token here-' \
  --header 'Content-Type: application/json' \
  --cookie
dtCookie=v_4_srv_1_sn_D28B3CDCC92313F57EE1BFF441B84D37_perc_100000_ol_0_mul
_1_app-3Aea7c4b59f27d43eb_1_rcs-3Acss_0 \
  --data '{
      "properties": {
            "Address": 1526,
            "Street": "W VERNON AVE",
            "Suffix": "",
            "Side": "Side",
            "Site": "0",
            "OnStreet": "S DENKER AVE",
            "FromStreet": "",
            "ToStreet": "",
            "UNIQUEID": "LE20200116081532",
            "NOTES": "Mayor Garcetti's demo tree planting.",
            "ChangeTIME": "12:26:23",
            "Inv_Time": "08:19:16",
            "ChangeDATE": "05-26-2021",
            "Inv_Date": "01-16-2020",
            "SPP": "Handroanthus heptaphyllus",
            "ParcelID": "111B149-120",
            "Observ": "None",
            "OHUtility": "Yes - Not Conflicting",
            "Clearance": "None",
            "Sidewalk": "On Grade",
            "Curb": "On Grade",
            "ISHB": "N/A",
```

Version: 2023.11.20

```
            "Pest": "No",
            "Defects": "Included Bark/Weak Union(s)",
            "Trunks": 1,
            "DBH": 1,
            "Height": "0 - 20 ft.",
            "Cond": "Good",
            "MaintPrim": "Training Prune",
            "Assessment": "No",
            "Council": "8",
            "GrowSpace": "Improved Parkway",
            "Neighbor": "Empowerment Congress Central",
            "Zone": "DRG Zone 1",
            "MtDistrict": "110 Southwest",
            "WellCover": "No",
            "SpaceSizeD": "3-4 ft",
            "SpaceSizeL": "N/A",
            "QCComp": "DRG Delivered",
            "TreeCage": "No",
            "INSPECTR": "LE",
            "PlantedBy": "",
            "PlantDate": 0,
            "StrSweep": "To Be Determined",
            "SectID": "1554200",
            "MaintRegn": "Bay Harbor",
            "DRGUpdated": "Yes",
            "HPriority": "None",
            "CMTree": "No",
            "CMTDate": 0,
            "CMTNotes": "",
            "WaterReq": "No",
            "D_Source": "DRG"
      },
      "geometry": {
            "type": "Point",
            "coordinates": [-13169595.893092, 4029269.4009002]
      },
      "relationships": {
            "parents": [],
            "children": []
      },
      "factype": "Street Sites"
}'
```


Version: 2023.11.20

HTTPS RESPONSE STATUS CODES

/sites/features HTTPS RESPONSE STATUS CODES

The following table outlines the various status codes and messages that may be received in response to a POST request sent to the api.treekeepersoftware.com/sites/featuers endpoint

Status Code	**Message**	**Meaning**	**Resolution**
200	Site, *remoteid*, is inactive, but is being uploaded.	The remote id which was submitted has previously been deleted.	A 200 OK response is sent to indicate that the feature was accepted, but no action was performed because the feature has been deleted.
200	The data for site, *remoteid*, is up to date already.	The feature with the submitted remote id as stored in the RowKeeper database is as new or newer than the one being uploaded.	The only way to force the upload to go through would be to increase the “ChangeTIME” property by at least one second
201	Site, *remoteid*, was successfully added to the RowKeeper database.	The feature was successfully added to the RowKeeper database.	—
202	Site, *remoteid*, was successfully updated in the RowKeeper database.	The feature’s data was successfully updated in the RowKeeper database.	—

400	The submitted data for site, *remoteid*, is too small. More data was expected.	The submitted JSON object is missing data	There are at least two required fields in a submitted JSON object: the Geometry (a GeoJSON-formatted representation of the feature) and the Properties (a struct containing the keys for all the attributes for the feature being uploaded)
403	various	There is an issue with authentication.	Verify that the submitted projectid, key, username, and password are all correct.
404	The facility for site, *remoteid*, was not found in the RowKeeper database.	The layer specified in the “factype” field of the submitted JSON does not exist in the RowKeeper database	Verify the spelling of the layer name.
406	The changeDate for site, *remoteid*, is zero.	The “ChangeDATE” property of the submitted feature is zero.	Fix the “ChangeDATE” property and upload again.
406	The change time for site, *remoteid*, is empty.	The “ChangeTIME” property of the submitted feature is empty	Fix the “ChangeTIME” property and upload again.
406	The INV_DATE for site, *remoteid*, is missing.	The “INV_DATE” property (inventory or creation date) is missing. This is a required field.	Make sure the “INV_DATE” property is included in the uploaded JSON data.

409	The Inv_Time submitted for site, *remoteid*, is not valid.	The “INV_TIME” property is not a valid time	Verify that the value in the “INV_TIME” property is, in fact, a time value.
409	The Inv_Date submitted for site, *remoteid*, is not valid.	The “INV_DATE” property is not a valid date.	Verify that the value in the “INV_DATE” property is, in fact, a date value.
409	One of the Work Areas submitted for site, *remoteid*, does not exist in RowKeeper	One of the values in the list included in the “WRK_AREA” property could not be found in the RowKeeper database	Verify that all of the work areas are spelled correctly.
409	The Work Area submitted for site, *remoteid*, does not exist.	The value included in the “WRK_AREA” property could not be found in the RowKeeper database	Verify that the work area is spelled correctly.
409	One of the Work Regions submitted for site, *remoteid*, does not exist in RowKeeper.	One of the values in the list included in the “WRK_REGION” property could not be found in the RowKeeper database	Verify that all of the work regions are spelled correctly.
409	The Work Region submitted for site, *remoteid*, does not exist.	The value included in the “WRK_REGION” property could not be found in the RowKeeper database	Verify that the Work Region is spelled correctly.
409	The ChangeDATE submitted for site, *remoteid*, is not valid.	The value submitted in the “ChangeDATE” property is not a valid date.	Verify that the last changed date is a properly formatted date. (yyyy-mm-dd)
409	The ChangeTIME submitted for site, *remoteid*, is not valid.	The value submitted in the “ChangeTIME” property is not a valid time.	Verify that the last changed time is a properly formatted time. (HH:mm:ss)
409	The Address submitted for site, *remoteid*, is not valid.	The address number is not numeric.	Verify that the correct address is being submitted for this feature.

409	*Value* is not a valid street for site, *remoteid*.	The value submitted for the Street, On Street, From Street, or To Street cannot be found in the RowKeeper database.	Check the spelling of the “STREET”, “ONSTREET/ONSTR” , “FROMSTREET/FROMSTR”, and “TOSTREET/TOSTR” properties in the submitted JSON and edit the street list if necessary.
409	The Suffix submitted for site, *remoteid*, is not valid.	The value in the “SUFFIX” property is too long	The Suffix can only be up to 5 characters long.
409	*value* is not a valid side for site, *remoteid*.	The value submitted for the side cannot be found in the RowKeeper database	Check the spelling of the “SIDE” property in the submitted JSON and edit the side list if necessary.
409	The Site submitted for site, *remoteid*, is not valid.	The value in the “SITE” property is not a numeric value.	Verify the “SITE” value.
409	*value* is not a valid species value for *property* for site, *remoteid*.	The value submitted for the specified property could not be found in the species list.	Verify the spelling of the submitted list value and edit the species list if necessary.
409	*value* is not a valid numeric value for *property* for site, *remoteid*.	The value submitted for the specified property should be numeric, but it is not.	Verify the correct value was submitted.
409	*value* is not a valid boolean value for *property* for site, *remoteid*.	The value submitted for the specified property is not a valid boolean value.	Valid values include “true”/”false”, “t”/”f”, “y”/”n”, “yes”/”no”
409	*value* is not a valid list value for *property* for site, *remoteid*.	The value submitted for the specified property could not be found in the appropriate list in the RowKeeper database.	Verify the spelling of the submitted list value and edit the list if necessary.

409	*value* is not a valid string value for *property* for site, *remoteid*.	The value submitted for the specified property could not be manipulated and saved to the RowKeeper database	Verify that the string is a valid string and that there are no non-unicode characters included in the string.
409	*value* is not a valid date value for *property* for site, *remoteid*.	The value submitted is not a numeric representation of a date (i.e.YYYYMMDD)	Verify that the date is being submitted in the format YYYYMMDD.
409	The type specified for *property* in the RowKeeper Database is invalid.	The data type for the specified property is not a valid data type in the RowKeeper database.	Contact DRG Support for more details and to resolve the issue.
409	There are geometry errors for site, *remoteid*.	There were errors when processing the submitted GeoJSON.	Verify that the GeoJSON for the submitted feature is properly formatted and is a valid geometry.
422	The locpark/locstreet data for site, *remoteid*, could not be written to the locparktable/locstreettable in the RowKeeper database.	There was an error when writing the location data (including the geometry) to the RowKeeper database.	Contact DRG Support for more details and to help resolve the issue.
422	The data for site, *remoteid*, could not be written to the sitetable in the RowKeeper database.	There was an error when writing the data for the feature to the database.	Contact DRG Support for more details and to help resolve the issue.
422	The multiple geog data for site, *remoteid*, could not be written to the RowKeeper database.	There was an error when writing the data for the multi-value location fields to the RowKeeper database.	Contact DRG Support for more details and to help resolve the issue.

422	The location data for site, *remoteid*, could not be updated in the locparktable/locstreettable in the RowKeeper database.	There was a database error trying to update the location data for the submitted feature.	Contact DRG Support for more details and to help resolve the issue.
422	the site data for site, *remoteid*, could not be updated in the sitetable in the RowKeeper database.	There was a database error when trying to update the RowKeeper database for the submitted feature.	Contact DRG Support for more details and to help resolve the issue.
500	Database query for Work Area failed to run for site, *remoteid*. Contact your project manager or DRG Support for assistance in resolving this issue.	There was a database error when attempting to query the database for the submitted “WRK_AREA.”	Verify that the work area is spelled correctly and then contact DRG Support to help resolve the issue.
500	Database query for Work Region failed to run for site, *remoteid*. Contact your project manager or DRG Support for assistance in resolving this issue.	There was a database error when attempting to query the database for the submitted “WRK_REGION.”	Verify that the work region is spelled correctly and then contact DRG Support to help resolve the issue.
500	There was an error setting up attributes for archiving.	There was a database error when beginning the archiving process before processing an update to the submitted feature.	Contact DRG Support for more details and to help resolve the issue.
500	Site, *remoteid*, was not successfully archived and therefore it was not updated.	The archive for the submitted feature was not successfully created.	Contact DRG Support for more details and to help resolve the issue.
500	There was a database error when archiving site, *remoteid*. So, it was not updated.	There was an error when attempting to write the archive for the submitted feature to the database.	Contact DRG Support for more details and to help resolve the issue.

Version: 2023.11.20

Keys description of the Uploaded JSON Object

Key	Purpose
Properties	The details of the record that is being uploaded, which is customized customer to customer and/or layer to layer. A specific schema describing these fields can be pulled using the data.treekeepersoftware.com API
Geometry	The Coordinates of the feature that is being uploaded, must be in SRID: 3857
Relationships	Sometimes records in one layer may be the child or parent of another layer. (i.e. Parcel is the parent of Tree). This relationship can be sent to our API using this object to specify the remote_ids of the parents/children involved
Factype	The layer (i.e. facility type) that the record is being uploaded.

JSON Object “Properties Key” Further Explanation:

This section details how to construct the JSON object that is being sent to us. It will be discreetly specified with the customer after award of the contract and final construction of the data schema that we are sharing between the two systems.

TK8/MRK can store a feature as a “Street” location or as a “Non-Street” location. Depending on how the feature is stored, requires different attributes to be passed. Most of the time, municipal software systems are *Street* location based and Utility Systems are *Non-Street* based.

Standard fields are required

Street Location Example		
Custom /Standard	**Key**	**Description**
Standard	Geometry.coordinates	The “geometry” object is a GeoJSON representation of the feature. The coordinates and type are a part of the GeoJSON standard. The coordinates will be in specific projection which is assigned to each TK8/MRK system SRID 3857

Version: 2023.11.20

Standard	Geometry.type	{Point, Linestring, Polygon, MultiLinestring, MultiPolygon}
Standard (Street)	properties.Address	Address number of the street address of the address. For example, for the address: 1600 Pennsylvania Ave The value stored in this property is “1600”
Standard (Street)	properties.Street	The street name of the street address. For example for the address: 1600 Pennsylvania Ave The value stored in this property is: “Pennsylvania Ave”
Standard (Street)	properties.Suffix	If an address has a “half number” or “letter” then that portion will be stored here. Ficticious addresses are commonly designated with an “X” in this field
Standard (Street)	properties.Side	Side of the property that the tree exists
Standard (Street)	properties.Site	Site number of the tree’s location relative to Address and Side - will be 1 through *N* based on the total number of trees present on each side of the property
Standard (Street)	properties.OnStreet	“On Street” for the tree.
Standard (Non-Street)	properties.geog1	Value from TK8/MRK’s Geography List Management for the appropriate layer, level 1 values. If List is not being used by the system then it must be set to “N/A”
Standard (Non-Street)	properties.geog2	Value from TK8/MRK’s Geography List Management for the appropriate layer, level 2 values. If List is not being used by the system then it must be set to “N/A”

Standard (Non-Street)	properties.geog3	Value from TK8/MRK's Geography List Management for the appropriate layer, level 3 values. If List is not being used by the system then it must be set to "N/A"
Standard (Non-Street)	properties.geog4	Value from TK8/MRK's Geography List Management for the appropriate layer, level 4 values. If List is not being used by the system then it must be set to "N/A"
Standard	properties.UNIQUEID	A unique identifier constructed by DRG field tools, TK8/MRK, or legacy software. Should be globally unique across the entire system. Currently, this is accomplished by creating the uniqueID as combination of Staff Initials, Date and Timestamp. DRG field protocol is each data collection user will have their own credentials with a unique set of initials. Legacy systems create the unique ID in a slightly different way with no built in meaning.
Standard	properties.NOTES	String represent general notes on the feature
Standard	properties.INSPECTR	The data collector initials. Default to "TreeAM"
Standard	properties.Inv_Date	Date of when feature was originally collected Stored as a string: "MM-DD-YYYY"
Standard	properties.Inv_Time	Time of when feature was originally collected Stored as string: "hh:mm:ss"
Standard	properties.ChangeDATE	Date of last feature edit Stored as string: "MM-DD-YYYY"
Standard	properties.ChangeTIME	Time of last feature edit Stored as string: "hh:mm:ss"
Custom	Properties.*SpeciesExample*	Species Attribute Will be validated against TK8/MRK species table, using the botanical name as the lookup key. If the corresponding

		value is missing in TK8/MRK, the record will be rejected, with a 409 response
Custom	Properties.*ListExample*	List Attribute Will be validated against TK8/MRK facilityattributelist table, using the facility attribute list value as the lookup key. If the corresponding value is missing in TK8/MRK, the record will be rejected, with a 409 response
Custom	Properties.*NumericExample*	Numeric Attribute Will only accept a numeric (or null) value, otherwise will return 409 response
Custom	Properties.*YesNoExample*	Yes/No Attribute Will only accept a 'Yes' or 'No' value, otherwise will return 409 response
Custom	Properties.*StringExample*	String Attribute Will accept any value and treat is a 'String'
Custom	Properties.*DateFieldExample*	Dates must be submitted as a numeric field in the format YYYYMMDD For Example, February 28th, 2022 would be submitted as: 20220228
Standard	type	Set to “Feature”
Standard	factype	What corresponding layer in TK8/MRK does this feature belong to. For purposes of this JSON Object: “Street Tree”

Specific Example JSON Object

Please note - this is purely an example. A specific example that applies to a particular customer and/or layer will be provided upon the production of the specific documentation for the customer's integration.


```
"properties": {
        "Address": 207,
        "Street": "BOX AVE",
        "Suffix": "",
        "Side": "In_Lot",
        "Site": "5",
        "OnStreet": "BOX AVE",
        "UNIQUEID": "Admin20231108111102",
        "NOTES": "",
        "ChangeTIME": "11:11:02",
        "Inv_Time": "11:11:02",
        "ChangeDATE": "11-08-2023",
        "Inv_Date": "11-08-2023",
        "SPP": "AILANTHUS ALTISSIMA",
        "CONDITION": "Dead",
        "DBH": 19,
        "MAINT": "Remove Tree",
        "Hazard": "Dead",
        "ROW": 0,
        "CurbROW": 0,
        "GrowSize": 99,
        "Date_Plant": 0,
        "Date_Remov": 0,
        "PIN": "",
        "Date_Inspe": 20231108,
        "Date_Stump": 0,
        "Date_Trim": 0,
        "BuffInsp": "ROSSH",
        "Inspect": "No",
        "Tag_Num": "",
        "Utilities": "No",
        "Parking": "NA",
        "District": "FILLMORE",
        "InvError": "No",
        "Status": "Unacceptable Existing",
```

```
        "Editing": "Buffalo",
        "2021update": "",
        "2022update": "No",
        "DRGUpdate": "Yes",
        "peja": "Yes"
    },
    "geometry": {
        "type": "Point",
        "coordinates": [
            -8775466.4647823,
            5298422.702469
        ]
    },
    "relationships": {
        "parents": [],
        "children": []
    },
    "factype": "Street Tree"
}'
```

Sending Work Data to TreeKeeper 8

Calling the API

Authentication/Authorization

Before using the API, you will need access to an Auth JWT token that denotes your user and project. Once you have that, you can include it in the request inside the following HTTP header.

```
Unset
Authorization: Bearer -jwt token here-
```

Upsert Endpoint

The purpose of this route is to upload new work records and update existing work records if their Work Change Date & Time are more recent than what is currently in the system. This route has the capability to upload/update multiple records in a single request.

```
POST https://api.treekeepersoftware.com/work_record_api/workrecord/upsert
```

In addition, there is a JSON request body where the details of the work record are included, so an appropriate HTTP header will need added:

```
Unset
Content-Type: application/json
```

Structure of Request Body

Within the first level of the JSON object there are two keys:

```
{
  "RequireExistingSiteRemoteID":"true",
  "Data": [
    {},
    {}
  ]
}
```

Version: 2023.11.20

RequireExistingSiteRemoteID:

The ‘RequireExistingSiteRemoteID’ key accepts ‘true’ or ‘false’ and defaults to true if unspecified. When set to ‘true’, the work record will only be uploaded if a specific existing Site_Remote_ID is included with the work record data, if Site_Remote_ID is not included, it will return an error.

When set to ‘false’, the work record will use the following methods to attempt to find a site to attach to, in the following order.

1. If X & Y coordinates are included, use those to find the closest site.
2. If Street Number & Name are included (and it is in a street type facility), use those to find the closest site.
3. If neither X/Y or Street Number/Name are included, use default Site Remote ID for facility. (this default is set in the database by request).

Work Record Data

The second key is the ‘Data’ key which is an array containing an object for each work record that will be uploaded in a request.

Required Keys for successful upload of work record

Field	Description / Notes
REQUIRED KEYS	
Work_Remote_ID	Accepts a string that will be used as that Work Records unique Identifier, coming from the remote system. This identifier will be used for all future interactions with this particular record in future API Calls such as updates.
Work_Change_Date	Accepts a date like “YYYY-MM-DD” to denote when the WR info was updated ***Note*** If intending to send an update for a work record via the API, then the Work_Change_Date and Work_Change_Time submitted will be compared to the record’s current date and time. If the record that is being sent is newer, then the API will process the

	update. Otherwise it will return a response indicating that it is up to date already.
Work_Change_Time	Accepts a time like “00:00:00.000” to denote when the WR info was updated ***Note*** If intending to send an update for a work record via the API, then the Work_Change_Date and Work_Change_Time submitted will be compared to the record’s current date and time. If the record that is being sent is newer, then the API will process the update. Otherwise it will return a response indicating that it is up to date already.
Work_Crew	Accepts a valid work crew from TK8 system.
Work_Priority	Accepts a valid work priority from TK8 system.
Work_Project	Accepts a valid work project from TK8 system.
Work_Type	Accepts a valid work type from TK8 system.
Work_Status	Accepts “requested”, “scheduled” or “completed”.
Work_Requested_Date	Accepts a valid date, only required when Work_Status = ‘requested’.
Work_Scheduled_Date	Accepts a valid date, only required when Work_Status = ‘scheduled’.
Work_Completed_Date	Accepts a valid date, only required when Work_Status = ‘completed’.
Optional Keys	
Work_Condition	Accepts a valid work condition from TK8 system. (If not specified, it will take this value from the related site’s condition).
Work_Species	Accepts a valid Common or Botanical species name from TK8 system. (If not specified, it will take this value from the related site’s species).
Work_DBH	Accepts a number or valid diameter list value from TK8 system. (If not specified, it will take this value from the related site’s DBH).

Work_Changed_By	Accepts name of valid user from TK8 system. (If not specified, will be the user that authenticated API request).
Work_Cost	Accepts a number. Represents cost (in $'s) of completing the work
Work_Time	Accepts a number. Represents the total time (in minutes) of doing the work
Work_Routing_Number	Accepts a number. Can be used for specifying the order that the jobs should be performed
Site_Remote_ID	Accepts a string representing Remote ID of site to attach WR to. (If RequireExistingSiteRemoteID = true, this becomes required.)
X	Represents Longitude of site in SRID 4326 projection.
Y	Represents Latitude of site in SRID 4326 projection. *Note* the X/Y columns will be used to find the "closest" tree if a Site_Remote_ID is not provided.
Street_Number	Accepts a number.
Street_Name	Accepts a valid street name in the TK8 system. *Note* the Street_Number and Street_Name field will be used to find a tree with a common address as the work record. The work record will be attached to a tree site at that common address.
Custom Work Attributes	
TBD	You can also include custom attributes in your request. Their key will be the Interface Name of the attribute which can be found in the Admin portal of TK8 under Work Attribute Management.
Comments	
Comments are included as an array, with each comment object containing an optional array of Tags. Booleans Show_Guest and Show_Contractor determine if guests and contractors can view the comment.	
Comment_Remote_ID	A unique ID for the comment

Tags	In TreeKeeper 8, comments can be tagged. The specific tag we should use for this integration is still to be determined
Comment	The comment itself
Show_Guest	True - guest users can see this comment False - guest users cannot see this comment
Show_Contractor	True - contractor users can see this comment False - contractor users cannot see this comment
Associated Call	
Optionally you can include a “Call” key including call details that will attach an associated call to that record. Note that if an existing call is already associated with the record, the request will update the call.	
Call_Remote_ID	The unique identifier for the call
City	Text
Email	Text
State	Text (2 letter abbreviation)
Reason	Reason code from TreeKeeper 8
Priority	Priority from TreeKeeper 8
Call_Date	Date of the call - formatted YYYY-MM-DD
Last_Name	Text
First_Name	Text
Home_Phone	Text
Caller_Classification	Caller Classification from TreeKeeper 8

API Response

After a request is sent to the API, you will receive a response with an object containing two keys, “Successes” and “Failures”. These two keys contain an array of objects representing failed

and successful work record uploads. Each object contains a status code representing the type of success/failure and a message containing the Work_Remote_ID and short error message.

```
{
  "Successes": [
    {
      "statusCode": 200,
      "message": "Successful upload of Test10"
    }
  ],
  "Failures": [
    {
      "statusCode": 400,
      "message": "Failed to upload record: Test6 - Uploaded Work Record Change Date/Time is older than current Change Date/Time of record"
    }
  ]
}
```

Example Object to upload

```
{
   "RequireExistingSiteRemoteID":true,
   "Data":[
      {
         "x":"0000000.00000000",
         "y":"0000000.00000000",
         "Call":{
            "call_remote_id": "TEST_CALL_ID_2",
            "city":"Los Angeles",
            "email":"johndoe@test.com",
            "state":"CA",
            "reason":"UFD MyLA311 SR",
            "priority":"2_Medium",
            "call_date":"2017-02-14",
            "last_name":"John",
            "first_name":"Doe",
            "home_phone":"111-111-1111",
            "caller_classification":"Unknown"
```

```
        },
        "work_crew":"UFD_TBD",
        "work_type":"UFD Unassigned",
        "Work_Layer":"Street Sites",
        "street_name":"100TH ST",
        "work_status":"completed",
        "work_project":"UFD_Unassigned",
        "street_number":3151,
        "Work_Comments":[
           {
              "Comment_Remote_ID": "TEST_COMMENT_ID_1",
              "Tags":[
                 "Pre Work"
              ],
              "Comment":"Tree Removal",
              "Show_Guest":false,
              "Show_Contractor":true
           },
           {
              "Comment_Remote_ID": "TEST_COMMENT_ID_2",
              "Tags":[
                 "Post Work"
              ],
              "Comment":"Tree Removed",
              "Show_Guest":false,
              "Show_Contractor":true
           }
        ],
        "work_priority":"2_Medium",
        "work_remote_id":"1-11111112",
        "Work_Change_Date":"2023-03-02",
        "Work_Change_Time":"09:47:06",
        "Work_Completed_Date":"2017-02-14",
        "Work_Requested_Date":"2017-02-14",
        "Work_Scheduled_Date":"2017-02-14",
        "site_remote_id":"AlKi20220204110032"
     }
  ]
}
```

Domain Values:

Current lookup values for Species, Streets, and List Values are stored here:

Version: 2023.11.20