Sending Data Updates to Tenstreet

API Guide (rev 10/2017)

Contents

Introduction

Tenstreet has the ability for customers to POST XML to us and update specific pieces of information on a subject/applicant already in our system. If the subject is not already in our system, we reject the POST update. The information that may be updated in Tenstreet today includes:

- Subject Status (New, Hired, etc.)
- Subject Worklist
- Subject Tagging (setting a new tag, updating an existing tag, unsetting an existing tag)

At a high level, the Update process works by first attempting to match provided information to a subject (that is, finding the subject you want to update in our system), validating that we have what we need to do the update, then actually performing the update. After processing your POST, Tenstreet responds with ACCEPTED or REJECTED in XML (see section titled POST Response below for an example of what this XML looks like) and a description of what occurred.

Agreements and Acknowledgements

Understanding the API

You are responsible for understanding the basics of xml and HTTP posting, and understanding the information contained within this document.

Debugging

You are responsible for parsing the Tenstreet response for debugging purposes. Failure to do so will result in data not being accepted into the system. If the posting type requires you to send a response back to our system, you are also responsible for sending that response, otherwise our system will not consider operations complete.

Logging

We expect you to keep both post and response logs of data that has been sent and received. **We expect this to be kept for at least 1 calendar year**. Tenstreet keeps the same data for 1 calendar year. This ensures the ability to audit all transactions from both sides.

Data Accuracy

If you have access to update multiple accounts (usually in the case of a job board), you also acknowledge that you will be responsible for posting the correct information to the correct account. Tenstreet will only validate that the passed Company ID and Password match, but will not attempt to validate that the information being sent belongs to the Company ID provided.

Support Requests

We want you to receive and send data to our system as your business use requires. We have a dedicated support group that handles requests about integrating your data with us at integrations@tenstreet.com.

If you need to contact us for support, please note that we are working toward getting you a response in the order it was received in the support queue, and that can take several elapsed working days. Sending more requests will just put your need further behind in the queue as we will have to handle them all separately. So please send one really well-formed request at a time.

This service is very popular, technical, and requires us to free up resources to answer your questions. But we *will* get to them.

When sending support requests, **please include**:

---

a) The Tenstreet XML response you are receiving if you are receiving one at all

b) your outbound or inbound ips (as relevant)

c) The URL you are posting to, as it is quite often that clients post to the development URL for their production data and get a response that says the company id is invalid, so make sure you are posting to the right environment.

d) The XML you are sending or receiving (as relevant), This speaks volumes to us, and will ultimately expedite your request. Do not send us PII data like an SSN. Just replace that with xxx-xx-1234 in the file

before you send it. Or delete it completely, it is not usually relevant to a support case.

e) A phone number, time zone, and time frame in which we can reach you. Often these issues are less overhead to resolve with a single phone call rather than multiple emails.

Authentication to the Tenstreet API

To authenticate your request, you must supply an authentication node in your xml. Please review the following table for a description of the node.

TenstreetData\Authentication

Node	Data Type	Required	Description
ClientId	Integer	Required	Provided by Tenstreet. This is your internal client id in the Tenstreet API system, not your Company ID
Password	String	Required	Provided by Tenstreet. This is a password generated specifically for using the Tenstreet API. It will be used for all API service, but nowhere else in the system.
Service	String	Required	The service you are requesting.

Example

The xml you need to use for this node looks like the follow and must be at the top of your file:

```
<Authentication>
   <ClientId>2</ClientId>
   <Password>987u34hng87asdh</Password>
   <Service> </Service>
</Authentication>
```

Considerations

PLEASE NOTE: This above is sample data. To receive your credentials, please contact integrations@tenstreet.com

Overview

Subject Match Process

Here's how the subject match process works. We preferentially look for matches on the following information (in this order):

1. **DriverId**: DriverId is Tenstreet's unique subject identifier. If you provide a valid DriverId for your company, that is the subject we update. If the DriverId you provide is not in your company, we will reject your POST, even if the other data might have matched to a driver. So don't provide a DriverId unless you know it's valid.

2. **Social Security Number** (SSN): If SSN is provided, we search on SSN. Note that all of the following matches (3 to 5 below) only take place if the SSN in the matched record is blank (else we'd have matched on it in this rule). For example, if the SSN is different, it doesn't matter if we match on email, we assume it's a different person.

3. **Email Address and Last Name**: If the email address and last name match, we update the record.

4. **Primary Phone and Last Name**

5. **Secondary Phone and Last Name**

If we do not have a match at this point, we reject the POST.

Validation

For several of the fields, we make sure that the data are valid. For example, if Status or Worklist are updated, we ensure the values provided are valid for your company. We also make sure that we have all of the fields we need for all updates. That is, we make an attempt to do all validation ahead of time so that you don't end up in a situation where one field is updated from the XML POST but another field fails. Said another way, either all updates should work or no updates will be processed.

Updating

We compare the value of the field you provide to the value of the field already in the table. Even if the values are the same (meaning, nothing was really updated since the values were already set to the data you provide), as long as validation succeeds, we will return an ACCEPTED response.

Application Data Updates

The following is an example of the various parts of data we allow a user to update. For application data, it must be inside the ApplicationData node:

```
<ApplicationData>

        <Status>New</Status>
        <Worklist>Queue 1</Worklist>
        <Address1>PO Box 149</Address1>
```

```
        <Address2>Appt 5b</Address2>
        <City>Tulsa</City>
        <State>OK</State>
        <Zip>74053</Zip>
        <PrimaryPhone>9181234562</PrimaryPhone>
        <SecondaryPhone>9181234569</SecondaryPhone>
        <InternetEmailAddress>bird@hunting.com</InternetEmailAddress>
        <WorklistHidden>yes</WorklistHidden>

</ApplicationData>
```

Tagging Updates

Tags are customer-specific data elements/fields that we store on a subject. Tagging is more complex than updating the Status or Worklist. This is because Tenstreet customer tags are customized for each customer. We not only have to match the data to a subject, we also have to match the tagging information you provide to a customer-specific tag. In the XML, the tagging section is referred to as CustomDataElements:

```
<UpdatedData>

        <CustomDataElements>
                <CustomDataElement>
                        <CustomDataPrompt>Termination Date</CustomDataPrompt>
                        <CustomDataValue>2012-01- 01</CustomDataValue>
                        <CustomDataId>TerminationDate</CustomDataId>
                </CustomDataElement>
                <CustomDataElement>
                        <CustomDataPrompt>Location</CustomDataPrompt>
                        <CustomDataValue>Tulsa</CustomDataValue>
                        <CustomDataId>Location</CustomDataId>
                        <CustomDataCustomerCode>Ttown</CustomDataCustomerCode>
                        <CustomDataTableId>3474</CustomDataTableId>
                </CustomDataElement>
        </CustomDataElements>

</UpdatedData>
```

Example of tagging update XML

We have three different ways to determine which tag you're attempting to update. We do three different searches.

1. CustomDataTableId – The most specific way to specify the tag. This is an integer ID that corresponds to a single tag (what we call the Tag ID).

2. CustomDataId – This is a cross-reference field that we use to look up the Tag ID. Generally, it will be a programming-friendly version of how the field appears in the system. For example, a date tag called "Hire Date" might have a CustomDataId set to hire_date or HireDate. The customer can choose the CustomDataIds for each field and we'll store them for later lookup (we can set these to whatever you want).

3. CustomDataPrompt – This is *exactly* what the user sees in the user interface when they're looking at a tag. If the tag is "Hire Date" then we would expect the value to be "Hire Date".

Once we find a matching tag, we stop searching and move to updating. This means that you could conceivably provide a CustomDataTableId that is in conflict with CustomDataId or CustomDataPrompt. We update on the first matching tag. Also, we must find a match on all tag elements provided (CustomDataElement XML elements) in order to update any data provided to us. This is part of validation and prevents us from updating some elements and not others. If we're unable to find a tag, we reject the entire POST and explain what went wrong.

Once we know which Tag ID we're working with, we look at the provided data to determine whether we're updating or unsetting the tag's value. Most of the time, you will provide the tag value directly in the CustomDataValue field. If the tag is a list in Tenstreet, you have the option of providing a cross-referenced value instead of the value as it appears in Tenstreet. The ability to update based on a cross-referenced value is unique to tags with dropdowns/lists and is optional.

In the example above, the Location tag provided a cross-referenced value (CustomDataCustomerCode) of "Ttown." If you have previously worked with us to set up the cross-referenced list values, we will insert a value of "Tulsa" into the tag value, even if you don't provide a CustomDataValue to us (our example above has both, but including both is optional). This is especially useful if you can only provide codes in your data, but need to have a human-readable value set in Tenstreet. Note that if you do not provide a CustomDataValue to us and we're unable to find a cross- referenced value, we will reject the POST. We will *not* set the value to the CustomDataCustomerCode. Further, if you were to provide both and the CustomDataCustomerCode derives to a different value than the provided CustomDataValue, the derived value wins and overwrites the CustomDataValue provided. Therefore, only use CustomDataCustomerCodes when you know there will be a match. Otherwise, just provide CustomDataValue.

If the tag is a date field in Tenstreet, we will validate the CustomDataValue. Dates must be in either MM/DD/YYYY or YYYY-MM-DD format. If you provide a value for a date tag that is not a valid date (and not blank, which would happen if you're unsetting the tag), we will reject the entire POST.

We identify the company in our system using the CompanyId field. Each company in our system has a unique ID (732, 10281, etc.). The sender would need to store these values for each company when they send them over.

You can fill in however few or many of the fields as you want, so long as the required fields are met.

Example

For an example of xml that can be sent, please click here

POST Response

You will always receive a POST response from Tenstreet, regardless if the attempted post was accepted or rejected, or if nothing occurred during the post. Please review the following table for a description of the XML and nodes that will be sent back.

TenstreetResponse

Node	Data Type	Description
Status	String	ACCEPTED or REJECTED. The description node will contain more information regarding this message
Description	String	Description of what happened
Version	Float	Version of the Tenstreet API that is responding
SourceIpAddress	String	The IP address the original request came from
CompanyPostedToId	Integer	Tenstreet company id that the data was posted to
CompanyPostedToName	String	Name in Tenstreet of company that data was posted to
DateTime	DateTime	Date and Time the response was sent
Mode	String	DEV or PROD. The mode of the system the response was sent from
DriverName	String	Name of the driver who's information was accepted

Of these fields, the two that are absolutely required are Status and Description. The rest are optional.

1. Status - ACCEPTED or REJECTED
2. Description - This can say anything, and is visible in the user interface if the status is Rejected (so that a user can take action). Some of our customers have listeners that require certain fields to be set and have validation routines that run, echoing responses in the description upon failure. We show these to the user to remind them to populate certain fields.

Example

For a sample XML file, please click here

Transmitting Data

Standard HTTPS POST

We've made an attempt to make the POST/response process as straightforward as possible. There are two types of data transmitting we accept. The standard HTTP POST and SOAP (Web Service) call.

Dev URL: https://devdashboard.tenstreet.com/post/
Production URL: https://dashboard.tenstreet.com/post/

We will provide your CompanyId to you, along with a ClientId and Password. POST the data as an XML string to us with text/xml headers (not urlencoded or with a POST variable). In procedural PHP, it would look something like this:

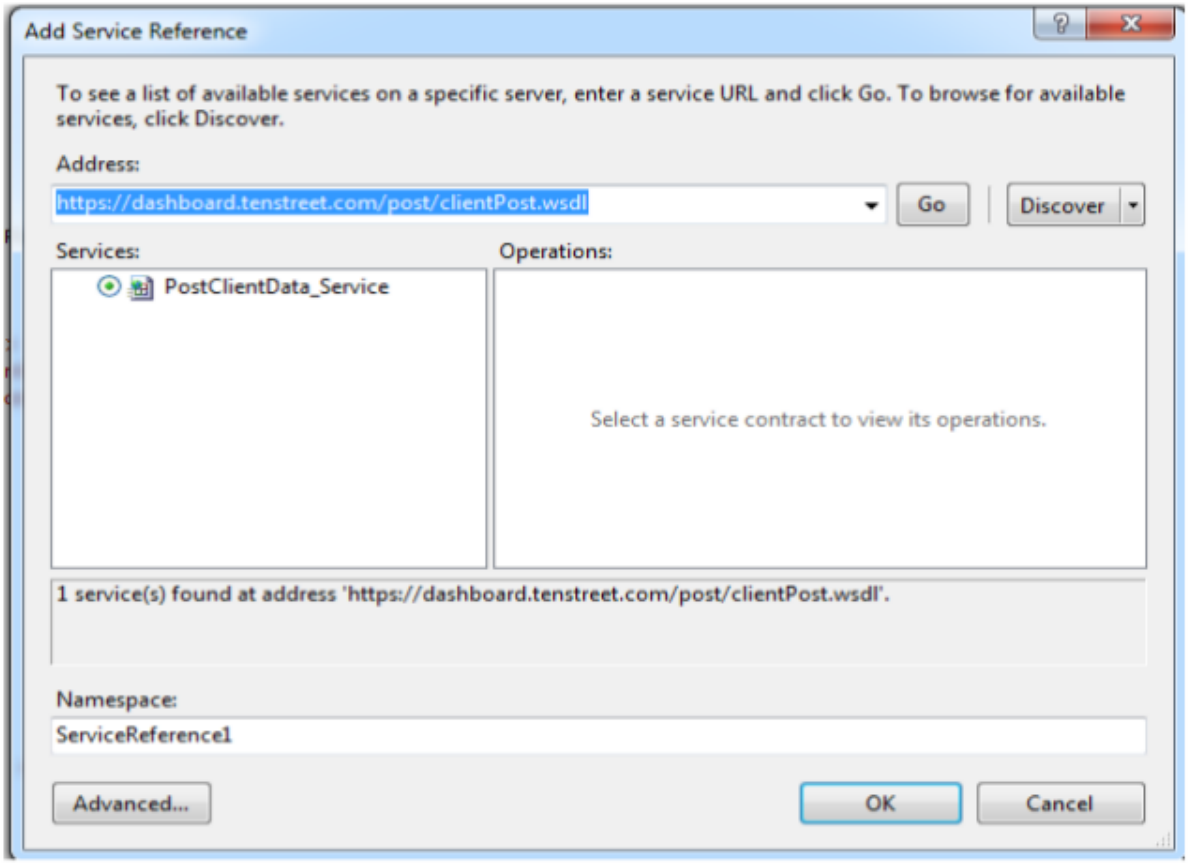
```
$post_address = 'https://devdashboard.tenstreet.com/post/';
$ch = curl_init();

curl_setopt($ch, CURLOPT_URL, $post_address);
curl_setopt($ch, CURLOPT_VERBOSE,  1);
curl_setopt($ch, CURLOPT_RETURNTRANSFER,1); // return into a variable
curl_setopt($ch, CURLOPT_HTTPHEADER, array('Content-Type: text/xml; charset=utf- 8'));
curl_setopt($ch, CURLOPT_POST, true);
curl_setopt($ch, CURLOPT_SSL_VERIFYPEER,  FALSE);
curl_setopt($ch, CURLOPT_SSL_VERIFYHOST, 0);
curl_setopt ($ch, CURLOPT_POSTFIELDS, $xml_data); // add POST fields

$response_xml = curl_exec($ch); // run the whole process

curl_close($ch); // Always close the connection

echo $response_xml; // $response_xml now contains Tenstreet response.
```

WSDL Consumption

1. The WSDL for consumption is located here:
   a. Prod: https://dashboard.tenstreet.com/post/clientPost.wsdl
   b. Dev: https://devdashboard.tenstreet.com/post/clientPost.wsdl

2. From Visual Studio (just taking a guess this is what IDE you are using), right-click on 'References'

3. Select 'Add Service Reference'



4. In the 'Address' box enter the WSDL location above and click 'Go':

5. Rename your 'Namespace' for your service reference, and then click 'OK'

6. Once the service has been loaded, calling the service and using the methods looks as follows:
   TenstreetPostService.ClientPost_PortClient client = new TenstreetPostService.ClientPost_PortClient();

   var xml = "XML from Appendix A Here"; var postResult = client.PostClientData(xml, **client_id**, "**Password**", "**service**");

The main points of interest are the first and last line. The first line is creating the web service client, and the last line is calling the generic 'PostClientData' method. Please note that VS will show you the arguments for this method, but they are 'xml', 'client_id', 'password', 'service'

The **client_id** is an integer Tenstreet will assign to you as soon as we are ready to do testing.

Password will be assigned to you by Tenstreet as well, and will serve as authentication and identification between our 2 systems.

Service will also be given to you by Tenstreet and will be a string.