Import from Point of Sale (IFPOS) API

This API is intended to be used under the following two situations:

1. **Immediately** after a sale is completed
2. **Immediately** after a past sale is modified

The API should not be used for batch uploads and is intentionally rate limited (will return a *429 Too Many Requests* message) to prevent this usage.

Calling the API

The API is available only through SSL/HTTPS and should be called using the POST method. The body of the request must match the IFPOSv1 JSON schema. The Content-Type must be *application/json* for your request and the request itself must have the dealer API key in it[1].

JSON Schema

The schema can be downloaded from here and use a tool like JSON Schema Validator to validate the body that will be sent via POST.

Performing Validation

1. Open https://cdn.etraxsales.com/resources/IFPOSv1-Schema.json in a tab
2. Open https://cdn.etraxsales.com/resources/IFPOSv1-SamplePayload.json in a tab
3. Open https://www.jsonschemavalidator.net in a tab
4. Copy & Paste the JSON from the step 1 tab into the left box
5. Right Box
    a. If you want to use the *sample* from step 2
        i. Copy & Paste the JSON from the step 2 tab into the right box
        ii. Replace `[YourAPIkey]` with *your API key* from TRAX[2]
        iii. If you adjust values on the right you will see validation information that lets you know if the request will be accepted
    b. If you would like to use your own JSON payload
        i. Copy & Paste your payload into the right box
        ii. Make adjustments as necessary to resolve any validation errors

API URL

The URL to POST the JSON to is: **https://api.etraxsales.com/v1/ifpos**

Pre-Integration Testing

You can use a tool like Postman to test calling the API before integrating it into your application:

---

[1] We have an alternative method available where the API key is part of the URL that is posted to, please contact us if you require this method instead.

[2] If you don't have an API key you will need a unique key for each TRAX Dealer you are integrating with, this key must be authorized by the Dealer and will be provided to the e-mail address they request us to deliver it to. A unique API key for IFPOS should be used independently of any API key already issued to the Dealer for their own use.

1. Download Postman from https://www.getpostman.com/apps
2. Install the downloaded application
3. Create an account and login to the application
4. Click **Request** on the **Create New** window that appears
5. Type **TRAX IFPOS** (or whatever you prefer) for the **Request name**
6. Save it to a collection of your choice (you may need to create one)
7. Change the method (shows *GET* by default) to **POST**
8. Copy & Paste **https://api.etraxsales.com/v1/ifpos** into the box that says *Enter request URL*
9. Click the **Body** tab under the URL
10. Click the **raw** radio button
11. Change the dropdown that defaults to **Text** to **JSON (application/json)** which will automatically put the appropriate *Content-Type* header in place
12. Copy & Paste the JSON payload you wish to send into the box
13. Click the blue **Send** button (your initial request will have a delay of up to 15 seconds)

API Response

The API will return a JSON structure which contains the following elements:

- **Status** – This is a HTTP status code, these can be the following:
  - 200 – OK – The data is now available in TRAX
  - 401 – Unauthorized – The API key was not successfully validated
  - 417 – Expectation Failed – Either the *ShowroomID* or *EmployeeID* is invalid[3]
- **ErrorDetails** – NULL if there were no errors otherwise this will be an array of strings which indicate what went wrong
- **AffectedRecords** – In an error situation this will be 0, in a success situation this will be 1 or 2. 1 indicates that this was a new sale record. 2 indicates that 1 record was removed and replaced with 1 record (thus 2).
- **ExecMS** – This is the amount of time in milliseconds it took us to do the work in our databases, this *will* be shorter than the amount of time that it took the method to run which includes validation, authentication, and scaling. If the method is called consistently it should return in ~300ms and the DB time will be around ~150ms – the first call in a day may take up to 15 seconds to complete due to some auto-magic scaling performed by our system (in other words: the more you use it, the faster it is).

[3] See the *ErrorDetails* collection for details