**Overview**

**High level process walkthrough:**

1. Get unprocessed spreadsheet from input folder
2. Get data from spreadsheet (VAT Number and Company Name)
3. Validate each row, checking the VAT number is valid and returned company name matches
4. If VIES API is unresponsive, manually search VIES
5. Update spreadsheet
6. Notify by email
7. Move spreadsheet to output folder

**Expected volumes and frequency:** Up to 8000 records once per week

**Current time to complete as is process manually:** 2 -3 hours

**Estimated time to complete automation:** 1-2 seconds per entry

**Platforms used:** (​<https://ec.europa.eu/taxation_customs/vies/>​)

**SOAP**: <https://ec.europa.eu/taxation_customs/vies/services/checkVATService>

**Design Considerations:**

Low coupling and modular workflow design is essential to allow for phase 2 development, substitution of input method, and substitution of validation service.

End user in process is waiting for validated data for payroll, so should be notified when process has been completed either successfully or unsuccessfully.

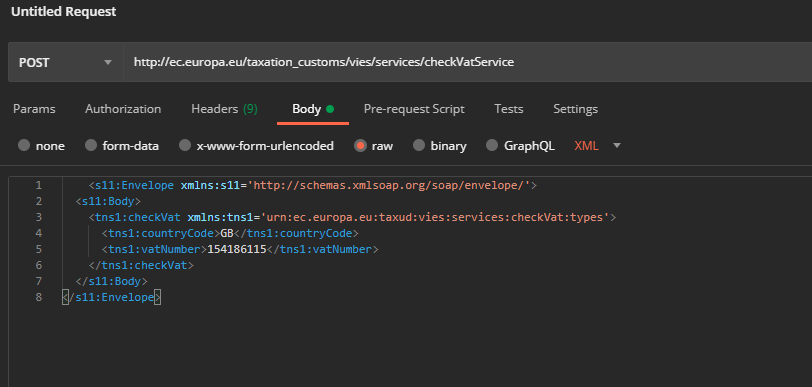
**Project dependencies (Custom Activities – source code provided):**

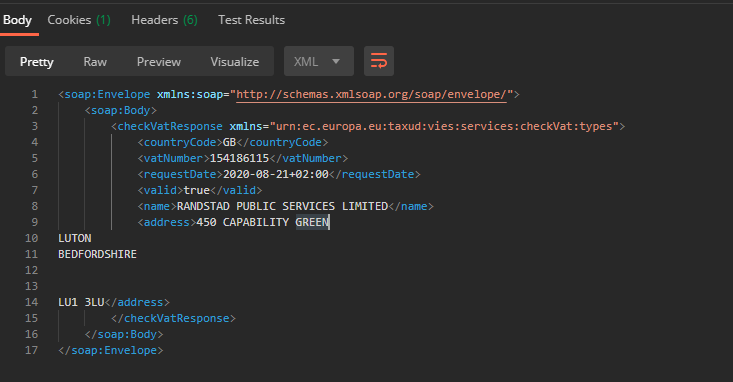
**UiPath.Activities.VIESValidation** – Handles the API call and returns validity and company name

**UiPath.Activities.CompanyNameMatching** – Takes the provided and response names and compares the two. Returns a Boolean for exact matches and 100% coefficient matches plus a percentage score for similarity based on Levenshtein edit distance (see assumptions).

**UiPath.Activities.ColourCode** – Returns red yellow or green colour names to use with system.drawing for highlighting rows depending on vat number validity, whether names were matched and the similarity score vs acceptable similarity threshold (recommended 85% plus, see below)

**Example of post request**:

**Example of response:**

**Typical Response Time:** 200 – 500 ms

**Note:**

* There are times where the VIES database is responding slowly, thus requiring more time to return a response. This should be considered while designing the automation

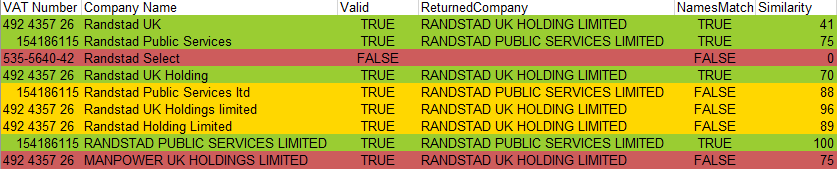
**Notes and Assumptions**

1. Since there is no column for country, it is assumed that the country code for validation will always be “GB-United Kingdom”.
2. Since no FTP etc is provided, input location is presently unknown.
3. Entry to Payroll System will or could be implemented as a phase two, but the MVP and main deliverable here is to validate spreadsheet data.
4. Invalid numbers will be manually queried, so any risk of not validating a valid entry are mitigated by the manual check, whereas wrongly validated entries are not. Any business exceptions should act as drop out points to manual working.
5. Third party batch validation services have been explored and ruled out either for data or security reasons and will not be used – VIES will remain the only endpoint for validating unless updated in future.

Specifically regarding name matching:

1. As per provided examples, names that are not exact matches but have 100% coefficiency (e.g. “Randstad UK Holdings” and “Randstad UK Holdings Ltd”) will be treated as matches with allowable omissions.
2. The robot should recognise that names are similar enough to potentially be the same company even if they cannot be matched in the above way. For example, “Randstad UK Holdings Limited” and “Randstad Holdings Limited” should be treated differently than two obviously different company names.
3. Name similarity will be measured in number of edits required to change one name to the other, which should highlight the level of error in entry. This will be converted to a percentage of the maximum possible difference. The Levenshtein algorithm will be provided as a custom activity – with more time other scoring algorithms could be implemented and compared as part of a decision engine.
4. The acceptable threshold for similarity must be configurable to allow balance between quantity of validated data and quality of validation.
5. To avoid confusion over the validation logic, a traffic light system will demonstrate where the robot has found an acceptable difference in unmatched names (see test output below).
6. Similarity thresholds can be configured – if in future the rest of the process is made manual and this decision making is monitored, it can be adjusted to set strictness of name matching.

Example Output with 85% similarity threshold setting:



**Possible enhancements**

Automate Payroll update process

Replace mouse clicks with Javascript injection on click events to prevent errors on virtual environments with no GPU capability.

Gsuite integration.

Expand name scoring decisions to consider alternative scoring and matching algorithms such as Smith-Waterman or fuzzywuzzy with python, explore business logic around name matching further.

Batch processing of spreadsheet for better restartability.

Fetch Excel spreadsheet from FTP or shared drive.