RMI Website Localization Project Requirements

# Jottings

* AutoLoc should only target ejs, js, vue, ts, tsx, csv, and json files inside the src folder and NOT files in dist folder. No other folder except src should be targeted and localized in the source code.
* AutoLoc needs to populate the extracted strings in JSON files stored in public/locales/[locale]/ folders
* AutoLoc should populate all ejs, js, vue, ts, tsx, csv, and json files strings in src/components/include/common folder in 1 JSON file (common.json) stored in public/locales/components/include/common/common.json
* AutoLoc should populate all ejs, js, vue, ts, tsx, csv, and json files strings in src/\_ejs/inc folder in 1 JSON file (common.json or inc.json) stored in public/locales/\_ejs/inc/common.json (or inc.json)
* We will not directly localize HTML files in dist folder anymore. We will only localize the ejs, js, vue, ts, tsx, csv, and json files in src folder and allow you to run your build process on src folder to generate the dist folder HTML files by yourselves later
* Therefore our job at RTranslate is to focus only on ejs, js, vue, ts, tsx, csv, and json files processing and NOT html (because you will build localized html from localized tsx and ejs files in src later)
* We have acknowledged your desire to output human-readable JSON keys and understand your choice of using AI to generate this feature. However, we may need to extensively test this approach for the desired outcome and inform you of its reliability. We will certainly use AI whenever it proves better than the "old fashion" ways.   
  Currently, we are testing both AI and the usual approaches and we will inform you of the findings in our subsequent meetings.
* With regards to following your bitbucket repo file structure, we will try to imitate the same folder patterns and please understand that we can only be systematic during automation. Except if you can specify the folders to target in the src in a list, then we can surely process the src folder as desired.

For L10n Format:

* TSX: { t('namespace.key') }
* TS: t('namespace.key') out of TSX
* EJS: ${{ namespace.key }}$
* VUE: {{ $t('namespace.key') }}
* JS: interpolation ${translate('key', langs)}, just value translate('key', langs) >> some modification may applied later
* CSV: I think it fine that from original CSV, multiple CSV in foreign languages are generated. In this case key separation is not necessary.

About Experiments on AI:

Preliminary observations on the AI/GPT L10n test I have been conducting this week entail the following:

Observations:

* Most files in each category (js, ejs, ts, tsx, and vue) tested caused max\_tokens limit error from the GPT-4 model which has a max\_tkens limit 8192 tokens.
* I have tried resolving this by first switching to a model version (gpt-4-32k) that can take a max\_tokens limit of 32,768 tokens, but some files still passed that limit in the input stream having on average 33,000 tokens. Some of the others had less but would cause the tokens limit error in the response stream where the model return the marked/localized code version or the JSON object from string extraction.
* The tokens limit error is based on the total number of tokens for (input + output) total number of tokens against the model's limit. It is hard to antecipate and regulate this limit except by chunking the file contents.
* I tried chunking on the files' contents but received mostly the same tokens limit error on some chunk returns with random/varying approach by the model.
* Chunking may also break original code due to limited control of the model's output in trying figure out the scope by itself.
* I want to try using LangChains for smooth chunking although I have worries on scoping and consistency assurance from it, to see if it will provide me with a smoother chunking experience.
* I have selected 20 files from each category of 5 file types, making 100 files to test on, however, I have been challenged with the max\_tokens limit error to complete this test smoothly. Some files passed the error but were not satisfactory in form/correctness.

Proposition:

* With your permission, I want pause the AI functionality test until I complete the first MVP of the AutoLoc app with conventional apprach. This will provide me with the chance to have a first release of the localized source code for the client on time while mitigating the desired AI approach simultaneously.
* Adjusting the ETA/delivery timeline of the app's MVP and first localized source code version for RMI until Sept 30 2024 or based on the client's review feedback, we adjust to accommodate additional feature implementations accordingly.
* AutoLoc is comprised of a UI (of various components) to manage a L10n project as per user requirements. It also comprise of various modules specializing each in a file type and desired approach and functionality.
* I would like to write a detailed Requirements Specifications document alongside the development progression, updating it based on the changes in development by client request on features. I would like to have the first UI design ready to start this Requirements Specifications document on confluence if possible or otherwise create a conceptual document for now and improve it moving on.