Unbabel Salesforce Challenge

Challenge made by: Unbabel

Solution developed by: Nuno Tomás

Date: 12/07/2019

Index:

Data Model

sObject : -Translation__c

-unbabelapi__Unbabel_Translation_Request__c

Fields (Translation__c):

- *From_Language__c;* (Text 50)
- *List_of_languages__c*; (PickList)
- *Original_Text__c;* (Text 255)
- *Status__c;* (Text 50)
- *To_Language__c;* (Text 50)
- *Translated_Text__c.* (Text 255)

Technical Scope

The purpose of this challenge was to build an application in Salesforce (either with Visualforce or Lightning) composed by a page where it was possible to input a text in a language and expect an output of another language selected. The Page was developed with Visualforce and tested with the Developer Console Tool.

Only the Admin user can see all of the translations made in the application, while a usual user can only see his translations. Nevertheless, everyone can use the application to the fullest potential.

The application uses two sObjects, the unbabel request and the translation, has four classes, the controller, the handler, the translated text and a test class translation test. Has one trigger which is the translated text class applied on the unbabel request sObject to be able to save the status and translated text on the desired fields.

There are one input field to put the text to be translated, two lists that contain the languages desired to translated and be translated to, respectively, and two buttons, one to submit the request and wait for and answer, and a deleteDB button to erase all of the translations made.

Additionally there is a list of requests made bellow the interface just for information purposes.

Use Cases

Actors:

User Admin with salesforce credentials;

Normal user with access to the application.

Cases:

Case1:

-The user logs in and goes to the Nuno Unbabel Challenge app;

-Chooses the tab “Translations”;

-Inputs a text to be translated, selects the language to be translated from, and the language to be translated to;

-The user clicks on the submit button;

Case2:

-The user already did the submit and got an answer;

-Clicls on the DeleteDB button;

Results:

Case1: If the request was successful the status of the request will be “Message Translated” and the result of the translated text will show on the table.

If the request wasn’t successful, the status of the request will be “Request Error” and no result will show.

Case2: The table of information with all the requests made will disappear and the list is empty now.

Missing requirements

Do too lack of time and knowledge, I wasn't able to create a pagination to the table of translations made, and I though it was worth mentioning beforehand since it was a requirement made. Same goes for the scalable application requirement.

Extras

The button to deleteDB was made for flow and utility purposes, which helps to have a lighter application.