***Design decisions:***

1) The reason to pick Angularjs is it does not need to build so the build steps for the submitted project are simple as compared to Angular 6.

Don't need extra plugins in Eclipse to first compile the transcript, copying javascript files to the right folder OR using webpack and then run the application.

Also, with 1-2 screens achieving SPA was easily possible in angularjs.

2) The reason to use a POST for submitting the request to the translation service instead of a PUT, is that every call to the webservice will be treated as a new file to be processed.

3) When the Web UI allows you to select the Input text file I have read its content and passed to the webservice.

The reason for taking the above approach instead of sending the file path to server, is that the local file accessible to the browse may not be accessible to Server.

4) I have added validations from the web ui and backend test cases for different scenarios. I am aware not all the scenarios might have been covered because of the wide range of cases that can occur with the text file and its content.

But I have tried to cover most of the scenarios.

5) The Markov Chain implementation expects the Text (read from text file), Prefix of key size (currently it is 2) number of words, suffix.

6) The scenario in which there are many suffixes it is a random selection for selecting the suffix.

***Possible improvements:***

1) More scenarios can be added to the test cases based on more scenario that will be uncovered based on the requirements.

2) Error messages showing above the fields can be moved to field specific error messages based on the requirements.

3) A weighted approach can be taken to select the suffix in case 1 prefix has many suffixes based on the requirements.