**SPELLING CORRECTOR**

**PROBLEM STAEMENT**

In this project we will be using deep learning approaches to build a spelling corrector.

**DESCRIPTION OVERVIEW**

A word needs to be checked for spelling correctness and corrected if necessary, many a time in the context of the surrounding words. A spellchecker points to spelling errors and possibly suggests alternatives. An autocorrector usually goes a step further and automatically picks the most likely word. In case of the correct word already having been typed, the same is retained.

There are different types of spelling errors.

1. **Non-word Errors**: These are the most common type of errors. You either miss a few keystrokes or let your fingers hurtle a bit longer. E.g., typing langage when you meant language; or hurryu when you meant hurry

2. **Real Word Errors**: If you have fat fingers, sometimes instead of creating a non-word, you end up creating a real word, but one you didn’t intend. E.g, typing buckled when you meant bucked. Or your fingers are a tad wonky, and you type in three when you meant there.

3. **Cognitive Errors**: The previous two types of errors result not from ignorance of a word or its correct spelling. Cognitive errors can occur due to those factors. The words piece and peace are homophones (sound the same). So you are not sure which one is which. Sometimes your damn sure about your spellings despite a few grammar nazis claim you’re not.

4. **Short forms/Slang/Lingo**: These are possibly not even spelling errors. May be u r just being kewl. Or you are trying hard to fit in everything within a text message or a tweet and must commit a spelling sin. We mention them here for the sake of completeness.

**TECHNOLOGY USE**

Here we will be using Anaconda Python 3.6 , Keras 2.2.4 using TensorFlow GPU 1.14.0 backend CUDA 10 with CuDNN 10

**INSTALLATION**

Installation of this project is pretty easy. Please do follow the following steps to create a virtual environment and then install the necessary packages in the following environment.

In Pycharm it’s easy

1. Create a new project.

2. Navigate to the directory of the project

3. Select the option to create a new new virtual environment using conda with python3.6

4. Finally create the project using used resources.

5. After the project has been created, install the necessary packages from requirements.txt file using the command pip install -r requirements.txt

In Conda also it’s eay

1. Create a new virtual environment using the command

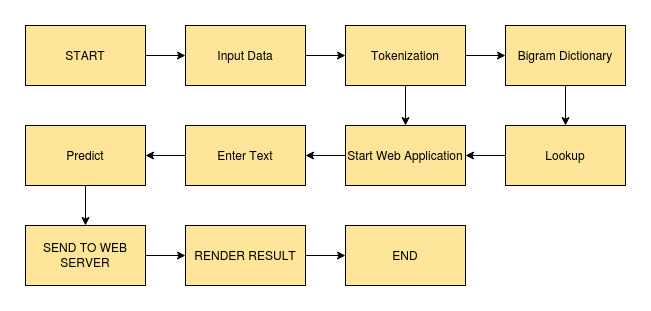
conda create -n your\_env\_name python=3.6

2. Navigate to the project directory.

3. Install the necessary packages from requirements.txt file using the command

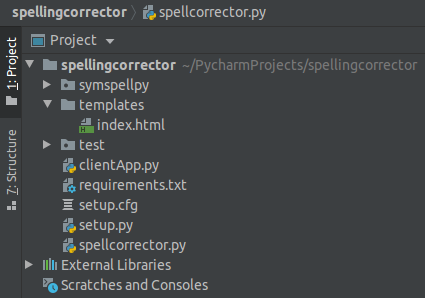
pip install -r requirements.txt

**WORKFLOW DIAGRAM**

****

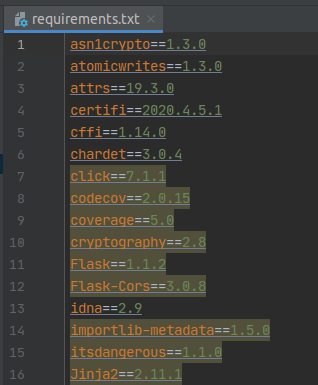
**IMPLEMENTATION**

**1. Project Directory**

****

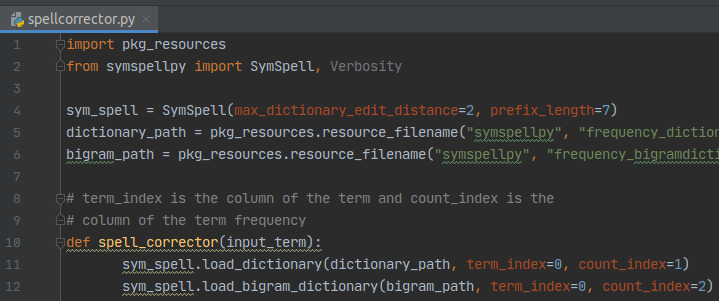
This above picture is of the project directory if we open the project folder using Pycharm. In the directory different types of files are there like for speliingcorrector, flask server etc.

**2. requirements.txt**

****

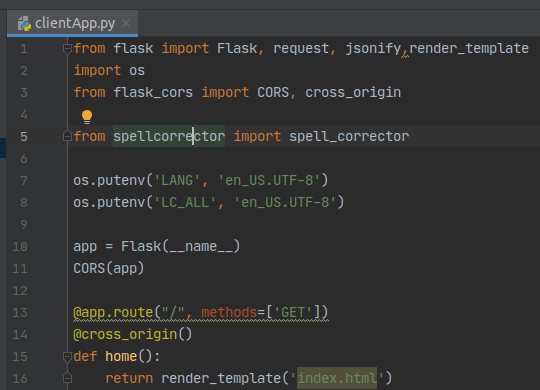
This file consists of the sequence to squence model architecture that has been built using Keras framework.

**3. spellingcorrector.py**

****

This file consists of the prediction process.

**4. clientApp.py**

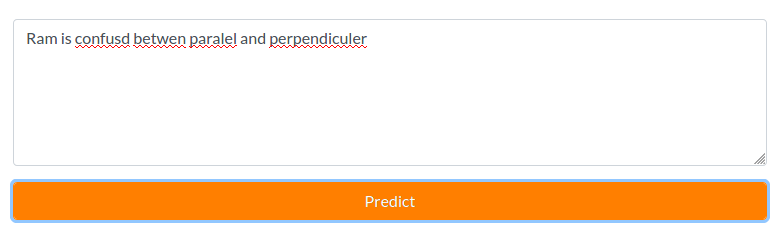


This file consists of flask and this is the entry point of application.

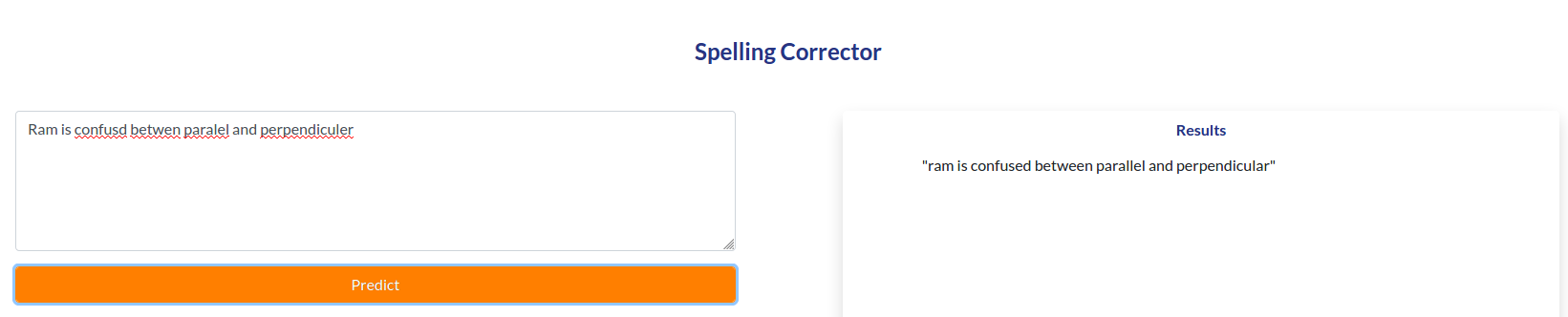
**TESTING IN LOCAL/API**

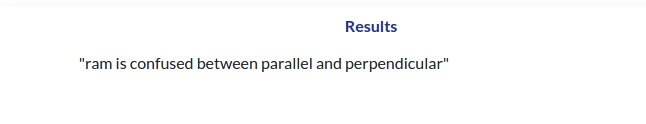
To do the test testing we need to run the clientApp.pyand after that web server will start at <http://0.0.0.0:5000/>

Enter the wrong spelled sentence and click on predict.



After clicking Predict



Finally the answers are shown in the results box.

**CONCLUSION**

Here we have successfully built a spelling checker and corrector which can be used to correct spelling if the user gives misspelled words.

**COMPARISION**

We can build better accurate models using pretrained models like BERT, GPT2 etc. Also here we have given very less training data but the user can increase the sizing of training data which will help to build a better model with better prediction.