**Prerequisites: Python Environment-**

Here is a list of Python libraries that need to be installed to your Anaconda path. Most of the commands are present in the .ipynb file in the “Environment Setup” section for these libraries, kindly execute these to install the needful.

* librosa
* os
* json
* tqdm
* math
* numpy
* pandas
* tensorflow
* matplotlib.pyplot
* sklearn
* copy
* soundfile
* sounddevice
* scipy.io.wavfile
* wavio

***Note***: *Make sure that those commands are modified if needed to install those libraries into your Anaconda path.*

**Steps To Create your Data-**

1. The user can run all of the code, and will be prompted to enter the folder/ directory path where the data will be stored. For eg. the user can specify Desktop’s path and the code will create a folder there.
2. The program will prompt the user to enter the following details. Kindly input these specific details.  
   1. **Total number of Words**: 9
   2. **Primary Keyword**: motivation
   3. **Secondary Keyword**: dedication, motion, monument, motorbike, notation, mutation, monsoon, moderator

***Note:*** *For the first time the user builds a model, use this set of words. However to test out the model further, you can pick your own primary and secondary words. In case you do so, pick those secondary words which sound similar to the primary keyword needed to be detected*

1. For recording audios, kindly make sure that you are in a silent environment with not much background noise. Use your laptop’s microphone (connected with your headphones if mic is present) to record the audio. The audio files to be recorded are 1.5s long, **so make sure that the keyword is completely present in these 1.5s.**
2. To record voice recordings, the program will prompt the user to ask if they are ready to speak. The user has to write a Yes. As they do so, the prompt **“Speak up!”** will come up on the screen. **Immediately enunciate** the given keyword as you see the above prompt. The user can go to the directory, play the audio file and verify the sample.
3. Repeat this for both primary and secondary words.

**Running the Model:**

Based on your created data, some augmentations will be applied and MFCCs will be extracted by the program and will be sent to the model which will be automatically created and run. Keep note of the Training and Testing accuracies separately.

***Note:*** *The training data (with augmentations) will be stored in the JSON\_PATH in the JSON file, so once you have created your data, load this JSON file each time to run the code and create a new model rather than recording data every time. The model will be stored as a .h5 file on the path, so once a model is made the same can be loaded in the code to perform manual testing as shown below.*

**Manual Model Testing:**

Now that a personalised Keyword Spotting Model is built, run this tester code to record any keyword and check its class label detected. For this, the program will once again prompt the user to speak a word and then it will classify your given word along with its probability of prediction for multi class classification.

The idea is to check if the primary word is being detected correctly. If the primary word is “motivation”, make sure that when “motivation” is enunciated, the program detects the presence of the primary word. Also, check that when a secondary keyword is spoken, it is not classified as a primary keyword. Keep these observations on your rough notebook or a document, containing your feedback if the model was able to detect the primary keyword properly.

***Note:*** *Once enough testing has been done, for manual model testing users can also speak in a keyword not present in our data which may sound similar to the primary keyword and check what class label is assigned to it. Save this in your feedback document.*