Configuration Manual

MSc Research Project
M.Sc. In Cybersecurity

Vivek Singh
Student ID: 21120315

School of Computing
National College of Ireland

Supervisor: Michael Pantridge

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name: ……. Vivek Singh………………………………………………………………………………………

Student ID: ………21120315……………………………………………………………………………………..……

Programme:………………M.Sc. in Cybersecurity ………… **Year:** 2022-2023

Module: ………………Academic Internship……………………………………….………

Lecturer: ……Michael Pantridge…………………….………

Submission Due Date: ……15/12/2022…………………………………………………………………………….………

Project Title: ……………Home automation framework through Voice recognition System for home security……………………………….………

Word Count: ……644………………… **Page Count:** ………………6…………..…….………

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: ……

Date: ……

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual

Vivek Singh
Student ID: X21120315

1 Introduction

The " Voice recognition System by analysing the different qualities of voice " configuration handbook describes how to configure the system's hardware and software, for this project MFCC and GMM techniques are being used for extracting the features form the voice and comparing the different voice based upon their log likelihood value. The code explained is to support the working of the project.

2 Dependencies

The following dependencies need be installed on the system in order for the application to be accessed and operate correctly on any of the systems:

1. Anaconda:
2. Python 3.6
3. TKinter
4. Scikit learn
5. Pillow
6. NumPy

3 Running/Accessing the program

Unzip the folder with the program file, now you can open the anaconda and activate the anaconda.

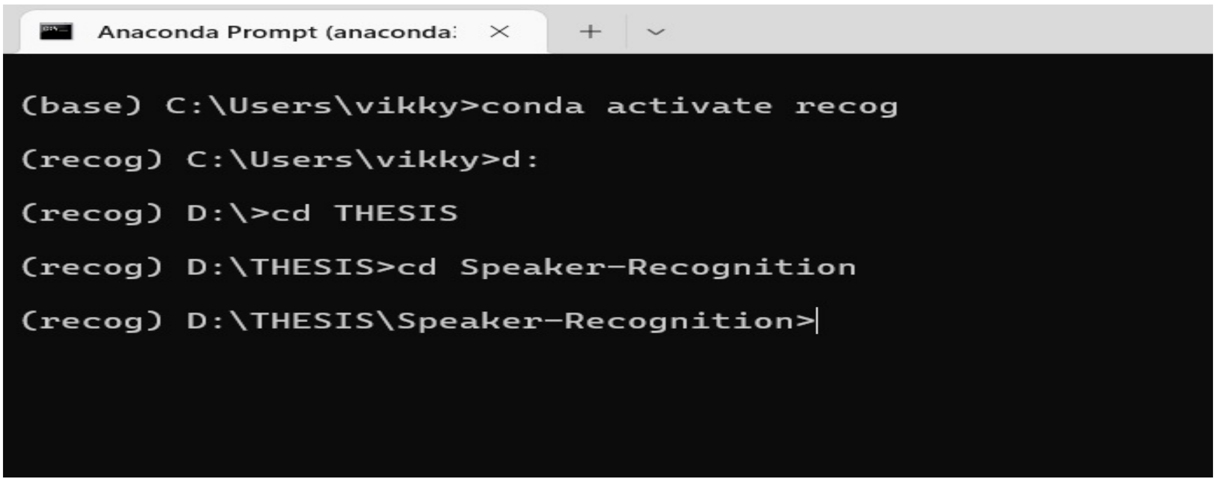

Figure 1:Starting the project

By typing the above shown command you can activate the anaconda. Now navigate to the Speaker-Recognition folder and run the homeapp.py file.

If you see the following image on the screen then the program is running properly.

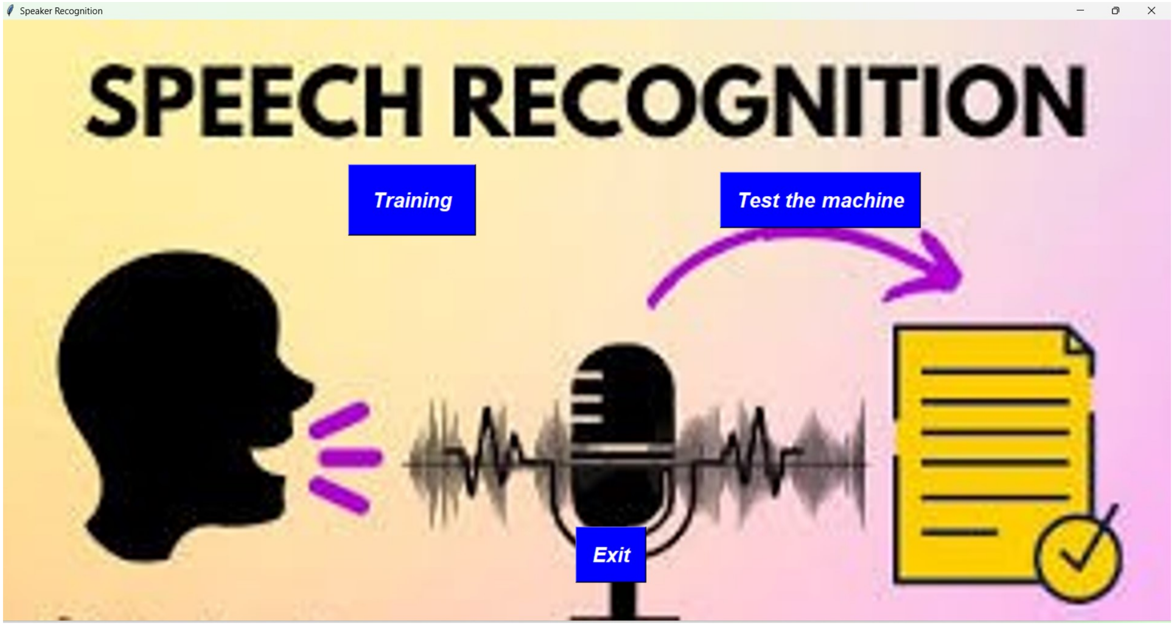

Figure 2:Front-end

4 Operation of the system

For training the machine, click on the train machine section and below window will appear.

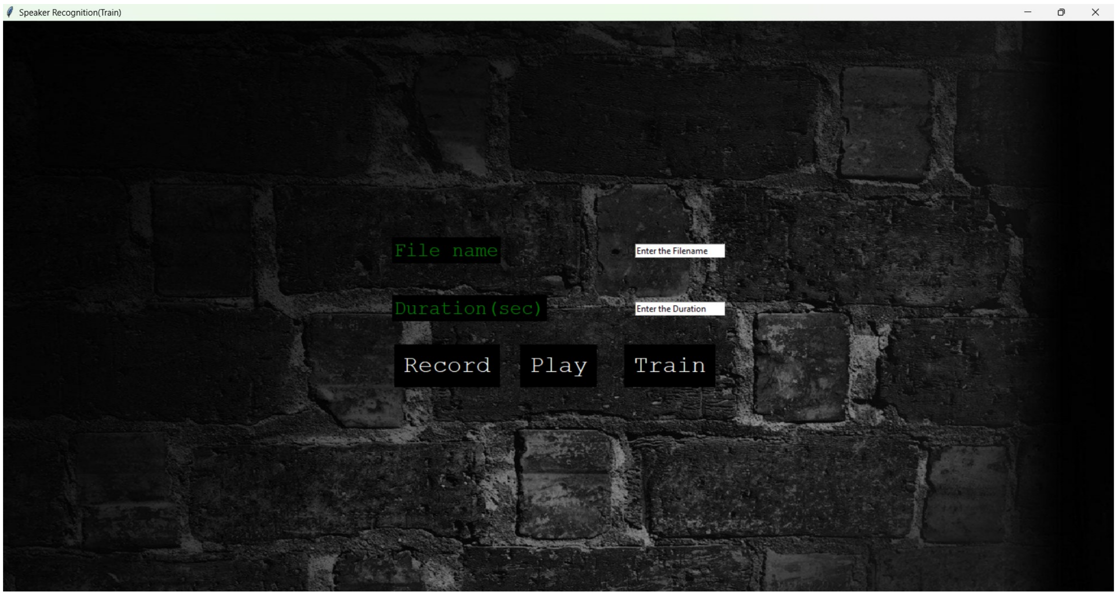

Figure 3:Training window

- For the purpose of this the name of the person and the duration of the audio should be written and,
- first 5 data sample to be recorded by clicking on the record button five times, the recorded data is being saved in training_data folder.
- For checking that the data is being recorded correctly you can simply play and check the audio as well.
- Now the data can be trained by clicking the train button, once the data is being trained you can close the window and move towards the testing section of the project.
- For testing the model, click on testing and then you can record the audio and test for the audio.

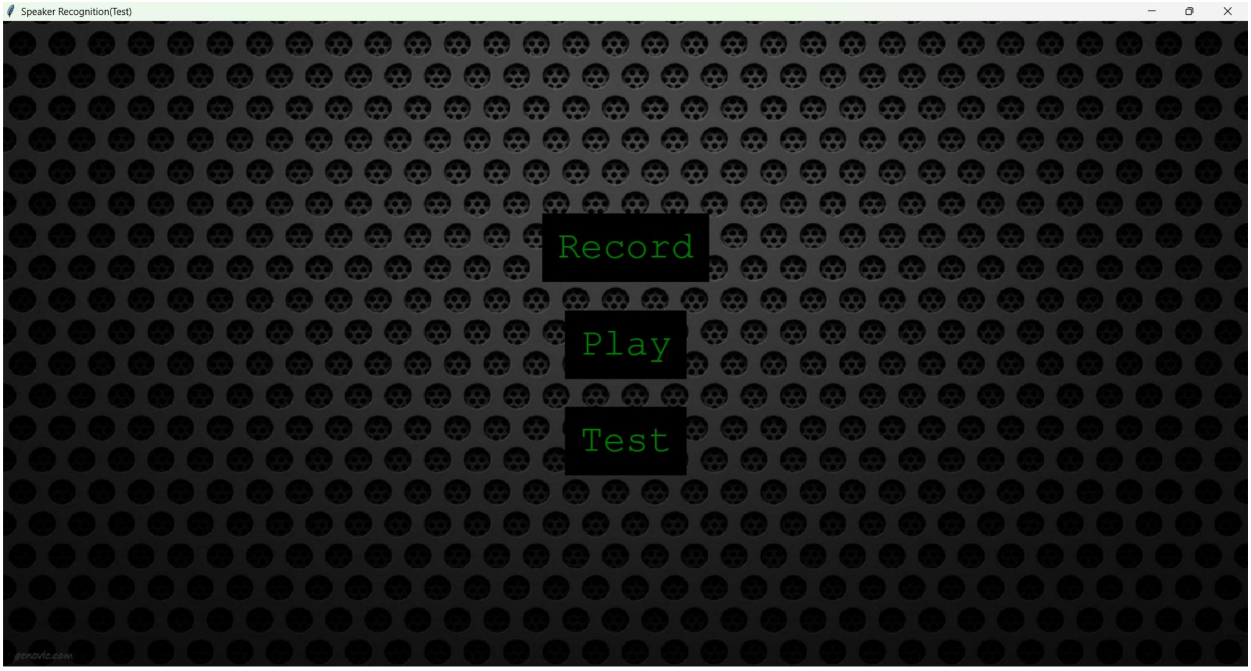

Figure 4:Testing window

5 Code Used:

The Mel-frequency cepstral coefficients (MFCC) is used in the project for the purpose of detection of speech and extracting the features for the voice and In a biometric system, a model called Gaussian mixture mode (GMM) is used to represent the probability distribution of continuous measurements or features.

```python
# -*- coding: utf-8 -*-
"""


"""
import numpy as np
import scipy
from python_speech_features import mfcc
from python_speech_features import logfbank
import scipy.io.wavfile as wav
from sklearn.mixture import GMM
import pickle
import os
import warnings
import mfeatures
warnings.filterwarnings("ignore")


def traine(x):
    file_paths=open("development_set_enroll.txt" ,'w')
    i=1
    while i<6:
        file_paths.write(x+str(i)+".wav\n")
        i+=1
    file_paths.close()

    #path to training data
    source   = r"training_data/"
    #path where training speakers will be saved
    dest = r"models/"
    train_file = "development_set_enroll.txt"
    file_paths = open(train_file,'r')

    count = 1
    # Extracting features for each speaker (5 files per speakers)
    features = np.asarray(())
    for path in file_paths:
        path = path.strip()
        # read the audio
        rate,sig = wav.read(source+path)
        mfcc_feat=mfeatures.extract_features(sig,rate)
        # extract MFCC

        if features.size == 0:
            features = mfcc_feat
        else:
            features = np.vstack((features, mfcc_feat))
        # when features of 5 files of speaker are concatenated, then do model training
        if count == 5:
            gmm = GMM(n_components = 16, n_iter = 200, covariance_type='diag',n_init = 10)
            gmm.fit(features)
```

Figure 5: GMM code

```python
from __future__ import division
import numpy
from python_speech_features import sigproc
from scipy.fftpack import dct
import scipy.io.wavfile as wav


def mfcc(signal,samplerate=16000,winlen=0.025,winstep=0.01,numcep=15,
         nfilt=55,nfft=2048,lowfreq=0,highfreq=None,preemph=0.97,ceplifter=22,appendEnergy=True,
         winfunc=lambda x:numpy.ones((x,))):
    feat,energy = fbank(signal,samplerate,winlen,winstep,nfilt,nfft,lowfreq,highfreq,preemph,winfunc)
    feat = numpy.log(feat)
    feat = dct(feat, type=2, axis=1, norm='ortho')[:,:numcep]
    feat = lifter(feat,ceplifter)
    if appendEnergy: feat[:,0] = numpy.log(energy) # replace first cepstral coefficient with log of frame energy
    return feat

def fbank(signal,samplerate=16000,winlen=0.025,winstep=0.01,
          nfilt=55,nfft=2048,lowfreq=0,highfreq=None,preemph=0.97,
          winfunc=lambda x:numpy.ones((x,))):
    """Compute Mel-filterbank energy features from an audio signal.
    """
    highfreq= highfreq or samplerate/2
    signal = sigproc.preemphasis(signal,preemph)
    frames = sigproc.framesig(signal, winlen*samplerate, winstep*samplerate, winfunc)
    pspec = sigproc.powspec(frames,nfft)
    energy = numpy.sum(pspec,1) # this stores the total energy in each frame
    energy = numpy.where(energy == 0,numpy.finfo(float).eps,energy) # if energy is zero, we get problems with log

    fb = get_filterbanks(nfilt,nfft,samplerate,lowfreq,highfreq)
    feat = numpy.dot(pspec,fb.T) # compute the filterbank energies
    feat = numpy.where(feat == 0,numpy.finfo(float).eps,feat) # if feat is zero, we get problems with log

    return feat,energy

def logfbank(signal,samplerate=16000,winlen=0.025,winstep=0.01,
          nfilt=55,nfft=2048,lowfreq=0,highfreq=None,preemph=0.97):
    """Compute log Mel-filterbank energy features from an audio signal.
    """
    feat,energy = fbank(signal,samplerate,winlen,winstep,nfilt,nfft,lowfreq,highfreq,preemph)
    return numpy.log(feat)

def ssc(signal,samplerate=16000,winlen=0.025,winstep=0.01,
        nfilt=55,nfft=2048,lowfreq=0,highfreq=None,preemph=0.97,
```

Figure 6:MFCC code