# Assignment\_1\_Report

## 1.The modifications to GUI and the codes

### 1.1 Set a button

Add a button to the bottom of the window on the basis of the original UI, and bind the slot function.

1. self.button = QtWidgets.QPushButton(self.centralwidget)
2. self.button.setGeometry(QtCore.QRect(170, 450, 50, 50))
3. self.button.setStyleSheet("QPushButton{border-image: url(img/1.png)}"
4. "QPushButton:hover{borderimage: url(img/1.1.png)}"
5. "QPushButton:pressed{borderimage: url(img/1.1.png)}")

### Binding slot function

Bind the slot function corresponding to the click signal of the set button, and the content of the function is to open the voice recognition thread.

1. application.ui.button.clicked.connect(clickButton)
3. **def** clickButton():
4. application.ui.gif.start()
5. t1 = threading.Thread(target=recognizer.recognizeSpeech, args=(application,))
6. t1.setDaemon(True)
7. t1.start()

## 2.Improve accuracy

In the process of recognition, I found that the accuracy of vocabulary recognition was not high, but words with similar pronunciation could be roughly recognized. Therefore, the optimization method of this project was to add similar words to the recognition database through continuous testing. If the identified words were in the database, the recognition would be considered successful.

1. **def** recognizeSpeech(application):
2. r = sr.Recognizer()
3. mic = sr.Microphone()
4. result = recognize\_speech\_from\_mic(r, mic)
5. choice = FuzzySearch(result['transcription'])
6. **if** choice:
7. **if** choice == 'music':
8. PlayMusic()
9. application.ui.gif.stop()
10. **if** choice == 'file':
11. OpenFile()
12. application.ui.gif.stop()
13. **else**:
14. **pass**

17. # 模糊搜索
18. **def** FuzzySearch(r):
19. a = r.split(' ')
20. b = ['machine', 'music', 'magic', 'magazine', 'major', 'mommy', 'market', 'supermarket', 'marriage', 'match',
21. 'maybe']
22. c = ['file', 'fire', ‘fight’, ‘fir’]
23. d = list(set(a).intersection(set(b)))
24. e = list(set(a).intersection(set(c)))
25. **print**(a, b, c, d, e)
26. **if** len(d):
27. **return** 'music'
28. **if** len(e):
29. **return** 'file'
30. **else**:
31. **return** 'file'

34. **def** PlayMusic():
35. # 播放音乐
36. # ShellExecute 查找与指定文件关联在一起的程序的文件名
37. # 第一个参数默认为0,打开，路径名，默认空，默认空，是否显示程序1or0
38. win32api.ShellExecute(0, 'open', r'goodnight.mp3', '', '', 1)

41. **def** OpenFile():
42. # 打开文件
43. win32api.ShellExecute(0, 'open', r'file.txt', '', '', 1)