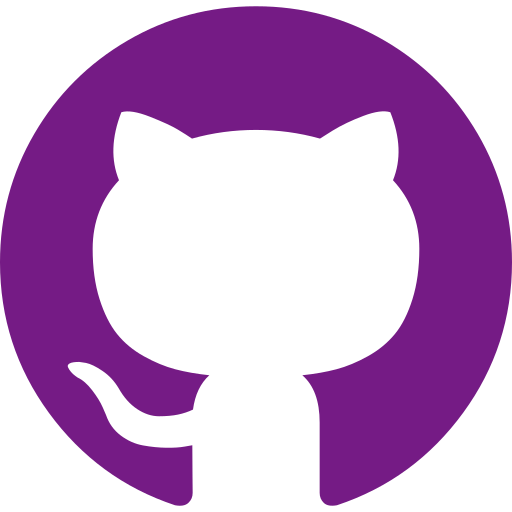
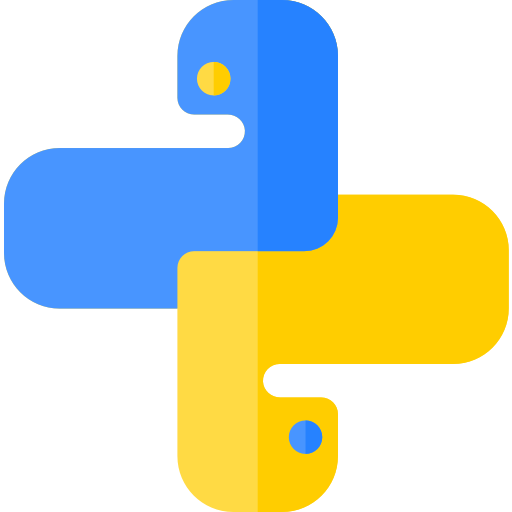
系統架構

 line-bot聊天機器人的架構簡單分為三段：line端、伺服器、本地端。當使用者對機器人輸入訊息時將會觸發webhook，並將訊息傳送至給定的webhook url、進入伺服器上的程式執行。程式執行結果再回傳至line使用者端。因此本組透過github在本地端共同開發，並使用免費平台heroku作為伺服器，再將github與heroku串連，使每次修改皆會在heroku上自動部署。

* line端：
  + 1. 註冊官方帳號、取得Channel Access Token以及Channel Secret、連結heroku url
    2. 由line @ manager製作功能選單
* heroku端：以github將包含主程式的資料夾上傳，資料夾中包含
  1. app.py (主程式)
  2. requirements.txt (使用額外的函式庫需在此注明，heroku將以pip方式逐行執行)
  3. Procfile (注明執行的py檔名以及使用的python版本)
* 本地端-程式架構：
  1. 使用者於功能選單點選功能回傳關鍵字
  2. 針對各關鍵字功能設計函數
  3. 將關鍵字與函數對應

琮仁心得：還記得大二上小傑老師第一次開商管程設，我興致勃勃地跑去加簽，很不幸的坐我左右的同學的雀屏中選，只有我落空。雖然這樣，趁著開學比較有空還是旁聽、寫作業到了第一次段考前，（然後就沒有然後了）。好不容易到了大四上再次有機會來學，有了以前的基礎再加上大四可以專心讀書的優勢，再次挑戰起來不像以前那樣吃力。Project剛開始的發想很雜亂，又是語音辨識又是數字辨識，還有反指標分析，不可勝數。後來聽了以前修課同學的分享，深深覺得無論做什麼，project做完的程式都要讓越多人可以使用越好。（萬分感謝上學期修課的柏樺告訴我可以用python寫line-bot）跟其他組不太一樣，老實說這份project中沒有遇到什麼太大的挫折或是debug的難題，就算flask, heroku什麼的都沒學過，但是看著網路上的教學一步步來大部分的問題都輕鬆地解決（有些太困難的就直接被放棄了XD）而linebot的特性也讓前後端的整合難度大幅降低，個人負責的餐廳推薦功能大約只花了12小時工時…，只能說選對工具真的能事半功倍XD

雖然期末報告結束了，這個project還沒有結束，自動辨識發票、連接資料庫自動對獎等等的功能希望可以在寒假期間繼續做下去。

能在畢業之前習得一項能夠把幻想在真實世界實踐的工具（的入門）真的是件幸福的事。